Expert Systems with Applications 38 (2011) 12312–12317

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Intelligent identification and control using improved fuzzy particle swarm optimization

Alireza Alfi *, Mohammad-Mehdi Fateh

Shahrood University of Technology, Faculty of Electrical and Robotic Engineering, Shahrood 36199-95161, Iran

ARTICLE INFO

Keywords:
Fuzzy particle swarm optimization
Parameter estimation
Genetic algorithm
Intelligent identification
Intelligent control

ABSTRACT

This paper presents a novel improved fuzzy particle swarm optimization (IFPSO) algorithm to the intelligent identification and control of a dynamic system. The proposed algorithm estimates optimally the parameters of system and controller by minimizing the mean of squared errors. The particle swarm optimization is enhanced intelligently by using a fuzzy inertia weight to rationally balance the global and local exploitation abilities. In the proposed IFPSO, every particle dynamically adjusts inertia weight according to particles best memories using a nonlinear fuzzy model. As a result, the IFPSO algorithm has a faster convergence speed and a higher accuracy. The performance of IFPSO algorithm is compared with advanced algorithms such as Real-Coded Genetic Algorithm (RCGA), Linearly Decreasing Inertia Weight PSO (LDWPSO) and Fuzzy PSO (FPSO) in terms of parameter accuracy and convergence speed. Simulation results demonstrate the effectiveness of the proposed algorithm.

1. Introduction

System identification is the first and substantial step to design a controller. According to an identified system model, a controller can be designed by means of various control methods to achieve the required specification. The least-squares approach is a basic technique often used for parameter identification. It has been successfully used to identify the parameters in the static and dynamic systems, respectively. However, it is only suitable for a structured model that being linearized in parameters. Once the model structure is not linear in parameters, this approach may be invalid (Astrom & Wittenmark, 1995). In addition, these techniques exhibit some fundamental problems due to their dependence on unrealistic assumptions such as unimodal performance landscapes and differentiability of the performance function.

To solve this problem, the heuristic optimization techniques seem to be more hopeful and promising alternatives to the traditional techniques. First, they do not rely on any assumptions such as differentiability, continuity or unimodality. Second, they can escape from local minima. Therefore, genetic algorithm (GA) (Cheng, Cheng, & Xie, 2010; Kömürcü, Tutkun, Özölçer, & Akpınar, 2008; Liao, 2009) and PSO (Lin, Chang, & Hsieh, 2008; Modares, Alfi, & Fateh, 2010; Sun, 2009; Zhao & Yang, 2009) were successfully used to estimate parameters for dynamic systems. Although the PSO has shown some important advances by providing high speed of convergence in specific problems, it exhibits some shortages (Modares et al., 2010). The Standard PSO (SPSO) has a poor ability to search at a fine grain because it lacks velocity control mechanism (Angeline, 1998; Clerc & Kennedy, 2002). To overcome this disadvantage, Shi and Eberhart (2001) used a fuzzy system to dynamically adapt the inertia weight namely Fuzzy PSO (FPSO). Consequently, the performance of PSO algorithm has improved well. This paper improves the FPSO and introduced a new PSO, namely Improved FPSO (IFPSO), to achieve a higher accuracy and convergence. The proposed IFPSO has two interesting characteristics: (1) to incorporate the difference between particles into PSO so that it can simulate a more precise biological model, the inertia weight is varied with the number of particles and (2) to truly reflect the actual search process, the inertia weight is set according to feedback taken from particles best memories.

The system identification not only can be used to consider the system characteristics but also the identified system can be applied as part of control law for instance to design the indirect adaptive control. This paper will also discuss an application of IFPSO algorithm to design an optimal Proportional–Integral–Derivative (PID) controller. Although most of industrial processes are complex nonlinear systems, they are still controlled with classical PID control structures, which are tuned to give good results around a fixed operating point. Under these circumstances, many PID tuning methods were proposed. The Ziegler–Nichols (ZN) method is an experimental one that is widely used, despite the requirement of a step input application with stopped process (Shinskey, 1996). One of the disadvantages of this method is the necessity of the prior knowledge regarding plant model. Once the controller was tuned by ZN method, a good but not optimum system response

* Corresponding author. Tel./fax: +98 273 3393116.
E-mail address: a_alfi@shahroodut.ac.ir (A. Alfi).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.04.009

will be reached. Many heuristic methods such as GA and PSO have recently received much interest for achieving high efficiency and searching global optimal solution in problem space (Visioli, 2001). In this paper, the PID controller gains are estimated by the proposed IFPSO such that the objective function is minimized.

In order to evaluate the performance of IFPSO, simulation results are finally given to demonstrate the effectiveness of proposed algorithm. The proposed algorithm has superior features in terms of stable convergence characteristics and good computational efficiency. Fast tuning of optimal PID controller parameters yields to high quality responses.

2. Nonlinear system identification

A technique is developed for identification a nonlinear systems. A class of discrete nonlinear systems is considered as follows:

$$\begin{aligned} x(k+1) &= f(k, x(k), u(k), P_1) \\ y(k) &= g(k, x(k), u(k), P_2) \end{aligned} \tag{1}$$

where $x \in R^n$ is the state vector, $u \in R$ is the input, $y \in R$ is the output, and P_1 and P_2 are vectors of system parameters. A model for system is introduced to identify unknown parameters with the same structure as system in the form of

$$\begin{aligned} \hat{x}(k+1) &= f(k, \hat{x}(k), u(k), \hat{P}_1) \\ \hat{y}(k) &= g(k, \hat{x}(k), u(k), \hat{P}_2) \end{aligned} \tag{2}$$

The same system input $u(k)$ is given to the system (1) and the identified model (2). The estimate parameter vectors are denoted using $\hat{P}_1$ and $\hat{P}_2$. Let $\theta = [\theta_1, \theta_2, \ldots, \theta_n]^T$ be a rearranging vector containing all parameters in $\hat{P}_1$ and $\hat{P}_2$, and n represents the number of unknown parameters.

The basic idea of parameter estimation is to compare the system responses with the parameterized model based on a performance function giving a measure of how well the model response fits the system response. In this study, the Sum of Square Error (SSE) for a number of given samples is considered as the fitness of estimated model parameters where the error is the difference between real and estimated responses. So, the fitness function is defined as follow:

$$\mathrm{SSE} = \sum_{k=1}^{N} [y(k) - \hat{y}(k)]^2 = \sum_{k=1}^{N} e^2(k) \tag{3}$$

where N is the number of given sampling steps, $y(k)$ and $\hat{y}(k)$ are the real and estimated values in each sample time, respectively. Our objective is to determine system parameters by using heuristic algorithms in such a way that the value of SSE is minimized and approaching zero as much as possible.

3. PID design

Many control problems can be handled very well by PID control (Astrom & Hagglund, 1995). The continuous PID control law is

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \dot{e}(t) \tag{4}$$

where e is a difference between the desired and actual outputs, u is the PID control law, parameters K_p, K_i and K_d are the proportional, integral and derivative gains, respectively. The discrete control law is obtained using trapezoidal approximations for Eq. (4) as

$$\begin{aligned} u(t) = u(k-1) &+ K_p[e(k) - e(k-1)] + K_i \frac{T_s}{2}[e(k) + e(k-1)] \\ &+ K_d \frac{1}{T_s}[e(k) - 2e(k-1) + e(k-2)] \end{aligned} \tag{5}$$

where T_s is a sampling period. How to adopt these three gains to meet the required performance is the most key in the PID control system. For simplification, let $\theta = [\theta_1, \theta_2, \ldots, \theta_n]^T = [K_p, K_i, K_d]^T$ be the control gain vector. The objective function for the PID controller is modified by

$$\mathrm{SSE} = \sum_{k=1}^{N} [y_r(k) - y(k)]^2 = \sum_{k=1}^{N} e^2(k) \tag{6}$$

where $y_r(k)$ is the desired output, and $y(k)$ is the system output. Fig. 1 illustrates a PID controller using heuristic algorithm.

4. IFPSO algorithm

Unlike population based evolutionary algorithms, PSO is motivated by the simulation of social behavior and each candidate solution is associated with a velocity. The candidate solutions, called "particles" then "fly" through the search space. In order to design the PSO algorithm, a population with a number of particles is created. Then, the velocity of every particle is constantly adjusted according to the corresponding particle's experience and the particle's companions' experiences. It is expected that the particles move towards better solution areas. The fitness of every particle can be evaluated according to the objective function of optimization problem. The velocity of every particle will be calculated at each iteration as follows:

$$v_i^{k+1} = \omega v_i^k + c_1 r_1 (pbest_i^k - x_i^k) + c_2 r_2 (gbest^k - x_i^k) \tag{7}$$

where in kth iteration, x_i is the position of the particle i, $pbest_i$ is the personal best position of this particle (memorized by every particle), $gbest$ is the global best position among all particles (memorized in a common repository), ω is the inertia weight, c_1 and c_2 are acceleration coefficients known as the cognitive and social parameters, respectively. Finally, r_1 and r_2 are two random numbers in the range [0 1]. The new position of every particle can be worked out as follows:

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{8}$$

The PSO algorithm performs iteratively until the goal is achieved. Although SPSO has shown some important advances by providing high speed of convergence in specific problems, it exhibits some shortages. It founds that SPSO has a poor ability to search at a fine grain because of lack of velocity control mechanism (Angeline, 1998). Many approaches are attempted to improve the performance of SPSO by variable inertia weight. The inertia weight is significant for the performance of PSO, which balances global exploration and local exploitation abilities of the swarm. A big inertia weight facilitates exploration; however it makes the convergence longer. Conversely, a small inertia weight makes the convergence fast; however it sometimes leads to local optimal. Hence, linearly decreasing inertia weight and nonlinearly decreasing inertia weight were proposed (Chang & Ko, 2009; Chatterjee & Siarry,

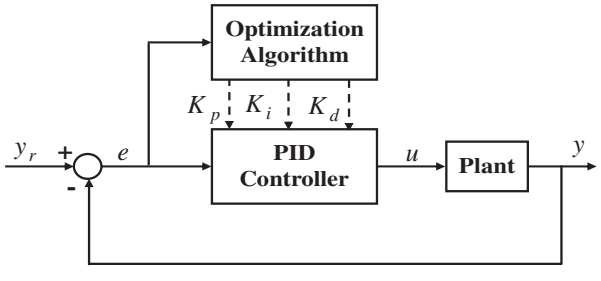

Fig. 1. PID controller design.

2006; Jiao, Lian, & Gu, 2008; Kennedy, Shi, & Eberhart, 2001; Ratnaweera, Halgamuge, & Watson, 2004; Yang, Yuan, Yuan, & Mao, 2007).

Nevertheless these algorithms improve the performance of PSO, they cannot truly reflect the actual search process without any feedback to know how far particle's fitness are from the estimated (or real) optimal value, when the real optimal value is known in advance. To overcome this shortage, Shi and Eberhart (2001) used a fuzzy system to dynamically adapt the inertia weight namely Fuzzy PSO (FPSO). Consequently, the performance of PSO algorithm has improved well. But, introducing the same inertia weight for all particles, by ignoring the differences among particles performances simulated a roughly animal background, not a more precise biological model, while the particles should be behaved differently according to their states. For instance, the particle which its fitness is far away from the real optimal value, a big velocity is still needed to globally search the solution space and thus its inertia weight must set to a large value. Conversely, for the particle which its fitness is close to the real optimal value only a small movement is needed and so inertia weight must set to a small value to facilitate finer local explorations. However, the same inertia weight is given to these opposite states.

Motivated by the aforementioned, this paper improves the FPSO by calculating the inertia weight for every particle according to the state of the particle. Consequently, every particle may have different trade off between global and local search abilities, since every particle locates in a complex environment and faces different situation. Due to this, a fuzzy logic is designed for every particle to provide the variations of weight factor as the output. The proposed fuzzy system has two inputs. The first input is called the normalized fitness of the current best position of ith particle (NFCBP$_i$). This input is determined as

$$\mathrm{NFCBP}_i^k = \frac{F(pbest_i^k) - F_{KN}}{F(pbest_i^1) - F_{KN}} \tag{9}$$

where $F(pbest_i^k)$ is the fitness of the best previous position of ith particle in kth iteration, F_{KN} is the known real optimal solution value and $F(pbest_i^1)$ is the fitness of the of ith particle in 1st iteration which is the worst acceptable performance of IFPSO for this particle. The second fuzzy input is the current value of the inertia weight factor for ith particle ω_i. The fuzzy output is variation of factor ω_i.

Each fuzzy variable has three membership functions namely small (S), medium (M) and large (L) as follows:

$$\mu_S(x) = \begin{cases} 1 & \text{if } x < x_1 \\ \frac{x_2 - x}{x_2 - x_1} & \text{if } x_1 \leqslant x \leqslant x_2 \\ 0 & \text{if } x_2 < x \end{cases} \tag{10}$$

$$\mu_M(x) = \begin{cases} 0 & \text{if } x < x_1 \\ \frac{x - x_1}{x_2 - x_1} & \text{if } x_1 \leqslant x \leqslant x_2 \\ 1 & \text{if } x_2 < x \end{cases} \tag{11}$$

$$\mu_L(x) = \begin{cases} 0 & \text{if } x < x_1 \\ 2\left(\frac{x - x_1}{x_2 - x_1}\right) & \text{if } x_1 \leqslant x \leqslant \frac{x_1 + x_2}{2} \\ 2\left(\frac{x_2 - x}{x_2 - x_1}\right) & \text{if } \frac{x_1 + x_2}{2} \leqslant x \leqslant x_2 \\ 0 & \text{if } x_2 < x \end{cases} \tag{12}$$

where μ represents the membership function, x_1 and x_2 are the crisp inputs. The fuzzy rules are given in Table 1.

Table 1
The fuzzy rules.

NFCBP	Inertia weight		
	S	M	L
S	L	S	S
M	L	M	S
L	L	M	S

5. Simulation results and comparison

This section demonstrates the feasibility of the IFPSO system identification and PID design. The simulation results are compared with those obtained from Real-Coded Genetic Algorithm (RCGA), Linearly Decreasing Inertia Weight PSO (LDWPSO) and Fuzzy PSO (FPSO). In all PSO algorithms, we set $c_1 = c_2 = 2$ and $V_{\max}$ and $V_{\min}$ are equal to the length of the search space (Chen, Qin, Liu, & Lu, 2005; Kennedy & Eberhart, 1995). In LDWPSO, ω decreases linearly from 0.9 to 0.4 (Shi & Eberhart, 1998). In addition, in RCGA, the reproduction probability P_r, the crossover probability P_c and the mutation probability P_m are set to 0.2, 0.3 and 0.2, respectively (Chang, 2007). To perform fair comparison, the same computational effort is used in all of LDWPSO, FPSO, IFPSO and RCGA. Thereby, the maximum generation G, population size S and searching range of the parameters in RCGA are the same as those in LDWPSO, FPSO and IFPSO. The membership functions for the fuzzy inputs and output in both FPSO and IFPSO are illustrated in Figs. 2–4.

In order to compare the performance of above algorithms, we identify an unstable nonlinear system (Chang, 2007)

$$\begin{aligned} x_1(k+1) &= \theta_1 x_1(k) x_2(k), \quad x_1(0) = 1 \\ x_2(k+1) &= \theta_2 x_1^2(k) + u(k), \quad x_2(0) = 1 \\ y(k) &= \theta_3 x_2(k) - \theta_4 x_1^2(k) \end{aligned} \tag{13}$$

where the real parameters are assumed $\theta = [\theta_1, \theta_2, \theta_3, \theta_4] = [0.5, 0.3, 1.8, 0.9]$.

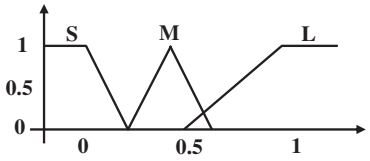

Fig. 2. Membership functions for NFCBP.

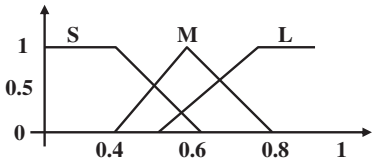

Fig. 3. Membership functions for inertia weight.

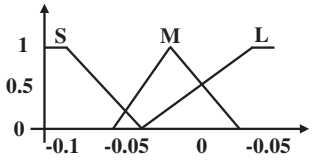

Fig. 4. Membership functions for change of inertia weight.

Table 2
A comparison on estimating parameters using RCGA, LDWPSO, FPSO and IFPSO.

	θ_1	θ_2	θ_3	θ_4	SSE
Real parameters	0.5000	0.3000	1.8000	0.9000	–
RCGA	0.4845	0.2981	1.7974	0.8795	0.7562
LDWPSO	0.5001	0.3002	1.8000	0.9002	5.6020×10^{-11}
FPSO	0.5000	0.3001	1.8001	0.9000	7.0236×10^{-14}
IFPSO	0.5000	0.3000	1.8000	0.9000	3.8636×10^{-22}

5.1. Parameter estimation

The objective of parameter identification is to determine θ as accurately as possible. The relative variables utilized in optimization algorithms are given by (Chang, 2007):

$$\theta_{1\min} = 0, \quad \theta_{1\max} = 2, \quad \theta_{2\min} = 0, \quad \theta_{2\max} = 2,$$
$$\theta_{3\min} = 0, \quad \theta_{3\max} = 2, \quad \theta_{4\min} = 0, \quad \theta_{4\max} = 2, \quad N = 8,$$
$$S = 20, \quad G = 200$$

The known optimal value F_{KN} is set to zero for the IFPSO. The optimization process is repeated 20 times independently. A comparison between RCGA, LDWPSO, FPSO and IFPSO is listed in Table 2. Figs. 5–8 show a great success of optimization process by using IFPSO algorithm as compared with other for the identified parameters θ_1, θ_2, θ_3 and θ_4, respectively. Moreover, the convergence of optimal SSE at each generation is plotted in Fig. 9. This comparison confirms the superiority of IFPSO algorithm in terms of accuracy and convergence speed without the premature convergence problem.

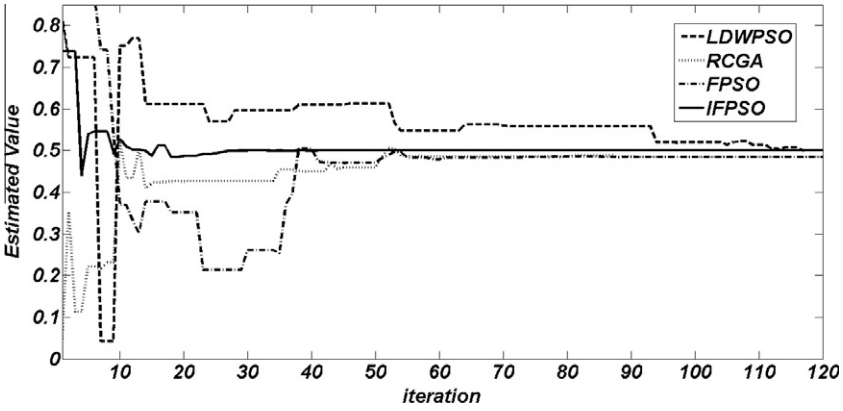

Fig. 5. A comparison on estimating θ_1.

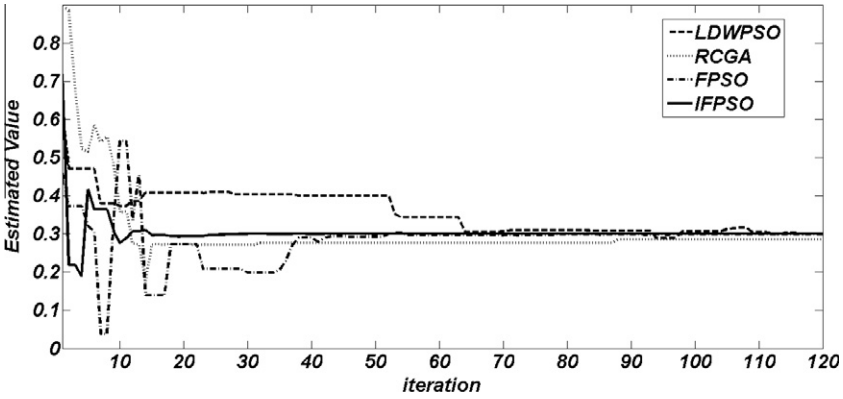

Fig. 6. A comparison on estimating θ_2.

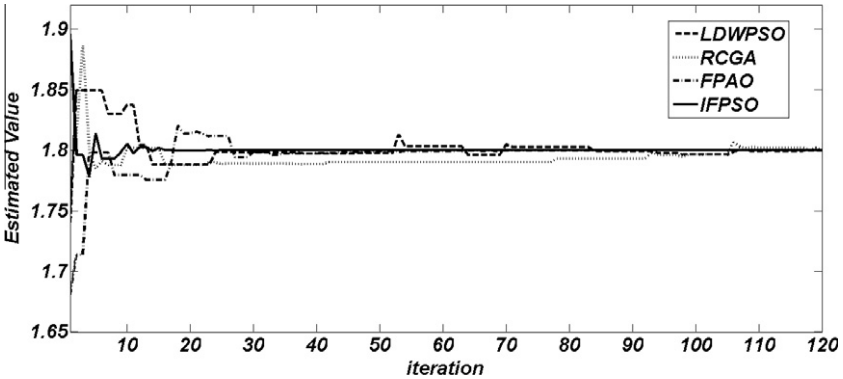

Fig. 7. A comparison on estimating θ_3.

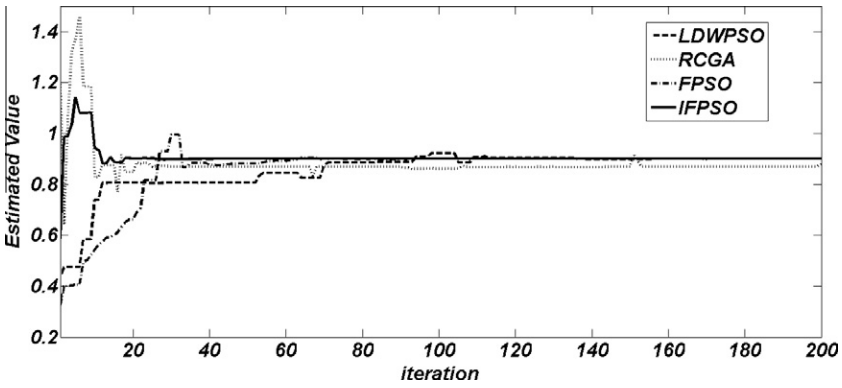

Fig. 8. A comparison on estimating θ_4.

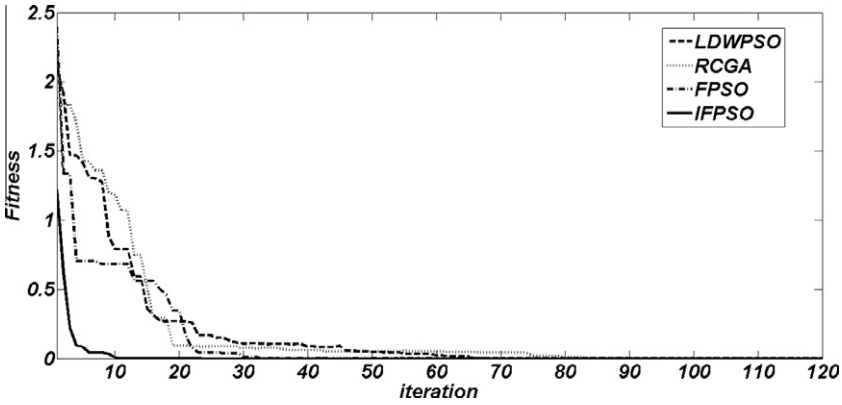

Fig. 9. A comparison on convergence of objective function.

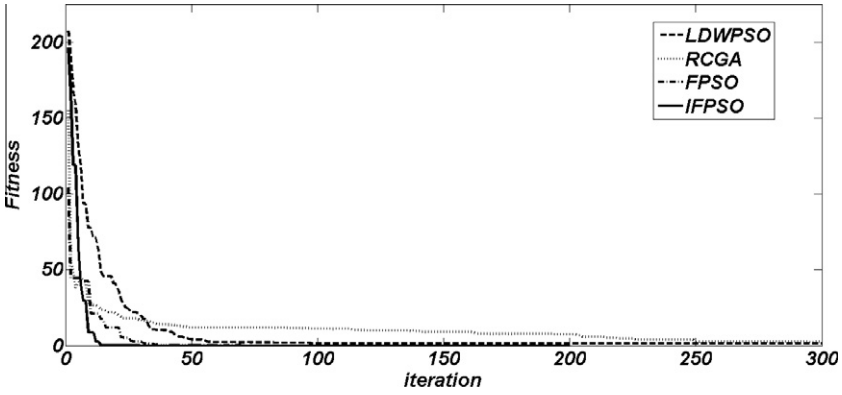

Fig. 10. A comparison on convergence of objective function for estimating PID gains.

5.2. PID controller design

Based on the above estimated results, a PID controller is designed for this system using the proposed IFPSO. In this simulation, the control objective is to minimize the difference between the plant output y and the desired output $y_r = 2$ in a regulating application. The population size S, number of sampling steps N and maximum generation G are considered as 20, 1000 and 300, respectively. Moreover, for the IFPSO, the known optimal value F_{KN} is zero. Fig. 10 represents a comparison on the convergence of SSE for obtained PID parameters using different algorithms. The initial search space for PID gains is defined as (Chang, 2007)

$$K_{p\min} = 0, \quad K_{p\max} = 1, \quad K_{d\min} = 0, \quad K_{d\max} = 1,$$
$$K_{i\min} = 0, \quad K_{i\max} = 1$$

The PID parameters as well as SSE for RCGA, LDWPSO, FPSO and IFPSO are given in Table 3. It is obvious that the proposed algorithm has superior features in terms of stable convergence characteristics, good computational efficiency and accuracy. Fast tuning of optimal PID controller parameters yields to high-quality solutions.

Table 3
A comparison on estimating PID parameters using RCGA, LDWPSO, FPSO and IFPSO.

	K_p	K_i	K_d	SSE
RCGA	0.8413	0.9932	0.0095	1.331
LDWPSO	0.8218	1.0307	0.0585	1.135
FPSO	0.8214	1.0105	0.0617	1.1102
IFPSO	0.8225	1.0044	0.0428	1.1032

6. Conclusion

In the present paper, a novel PSO algorithm has been introduced for identifying system parameters and PID controller gains namely IFPSO algorithm. In IFPSO, to incorporate the difference between particles into PSO, the inertia weight is varied with the number of particles. Due to this, for every particle, the fitness of its personal best is considered as an input to the fuzzy system for calculating the variation of its inertia weight. As a result, the IFPSO algorithm has a faster convergence speed and a higher accuracy. The proposed algorithm can simulate a more precise biological model and truly reflect the actual search process by setting the inertia weight according to feedback taken from particles best memories. The simulation results obtained from RCGA, LDWPSO, FPSO and IFPSO algorithms were compared. They have shown the superiority of the proposed IFPSO algorithm in identifying system parameters and PID control gains.

References

Astrom, K. J., & Wittenmark, B. (1995). *Adaptive control*. Massachusetts: Addison-Wesley.

Angeline, P. J. (1998). *Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences*. *Lecture notes in computer science* (Vol. 1447). London: Springer-Verlag (pp. 601–610).

Astrom, K. J., & Hagglund, K. J. (1995). *PID controllers: Theory design and tuning*. International Society for Measurement and Control.

Chang, Y. P., & Ko, C. N. (2009). A PSO method with nonlinear time-varying evolution based on neural network for design of optimal harmonic filters. *Expert Systems with Applications, 36*, 6809–6816.

Chang, W. D. (2007). Nonlinear system identification and control using a real-coded genetic algorithm. *Applied Mathematics and Computation, 31*, 541–550.

Chatterjee, A., & Siarry, P. (2006). Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization. *Computer and Operation Research, 33*, 859–871.

Chen, J., Qin, Z., Liu, Y., & Lu, J. (2005). Particle swarm optimization with local search. In *Proceedings of the IEEE international conference on neural networks and brain* (pp. 481–484).

Cheng, C. H., Cheng, P. J., & Xie, M. J. (2010). Current sharing of paralleled DC–DC converters using GA-based PID controllers. *Expert Systems with Applications, 37*(1), 733–740.

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions of Evolutionary Computation, 6*(1), 58–73.

Jiao, B., Lian, Z., & Gu, X. A. (2008). Dynamic inertia weight particle swarm optimization algorithm. *Chaos Solitons & Fractals, 37*, 698–705.

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the IEEE international conference on neural networks* (pp. 1942–1948).

Kennedy, J., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence*. San Francisco: Morgan Kaufmann Publishers.

Kömürcü, M. I., Tutkun, N., Özölçer, I. H., & Akpınar, A. (2008). Estimation of the beach bar parameters using the genetic algorithms. *Applied Mathematics and Computation, 195*(1), 49–60.

Lin, Y. L., Chang, W. D., & Hsieh, J. G. (2008). A particle swarm optimization approach to nonlinear rational filter modeling. *Expert Systems with Applications, 34*, 1194–1199.

Liao, H. C. (2009). Using GA to dispatch the obtaining quantity for minimizing the total cost based on consideration of patient safety in HSCM. *Expert Systems with Applications, 36*, 11358–11362.

Modares, H., Alfi, A., & Fateh, M. M. (2010). Parameter identification of chaotic dynamic systems through an improved particle swarm optimization. *Expert Systems with Applications, 37*, 3714–3720.

Ratnaweera, A., Halgamuge, S. K., & Watson, H. C. (2004). Selforganizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions of Evolutionary Computation, 8*(3), 240–255.

Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *Proceedings of the seventh annual conference on evolutionary programming, New York* (pp. 591–600).

Shi, Y., & Eberhart, R. C. (2001). Fuzzy adaptive particle swarm optimization. In *Proceedings of the congress on evolutionary computation* (pp. 101–106).

Shinskey, F. G. (1996). *Process control system: Application design and tuning*. McGraw-Hill.

Sun, T. H. (2009). Applying particle swarm optimization algorithm to roundness measurement. *Expert Systems with Applications, 36*, 3428–3438.

Visioli, A. (2001). Tuning of PID controllers with fuzzy logic. *Control Theory and Applications, 148*(1), 1–8.

Yang, X., Yuan, J., Yuan, J., & Mao, H. (2007). A modified particle swarm optimizer with dynamic adaptation. *Applied Mathematics and Computation, 189*, 1205–1213.

Zhao, L., & Yang, Y. (2009). PSO-based single multiplicative neuron model for time series prediction. *Expert Systems with Applications, 36*, 2805–2812.