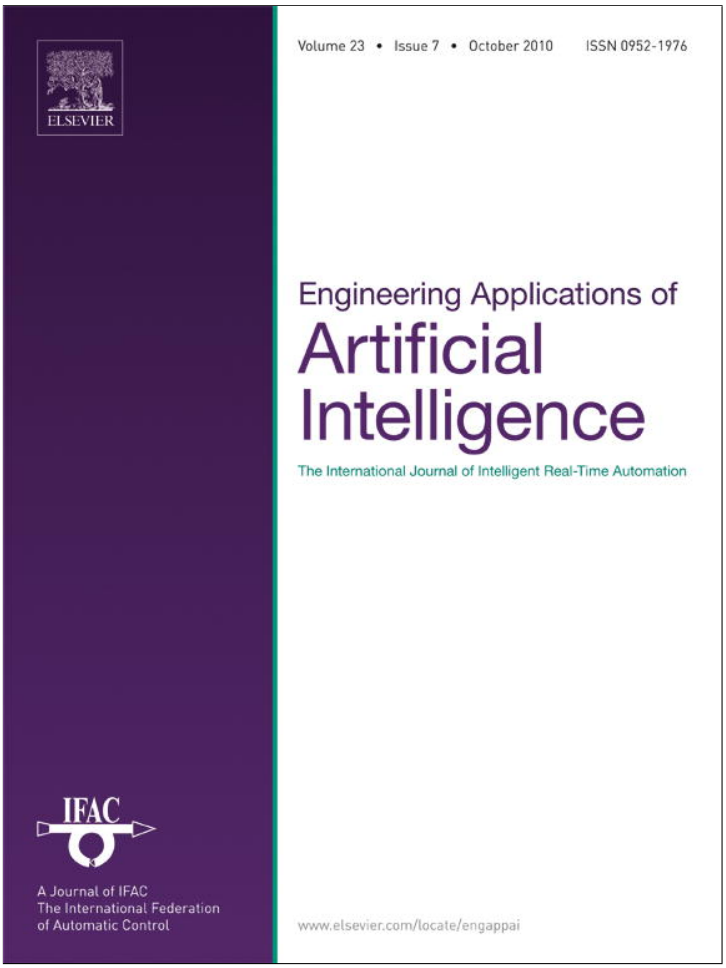

Engineering Applications of Artificial Intelligence 23 (2010) 1105–1111

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Parameter estimation of bilinear systems based on an adaptive particle swarm optimization

Hamidreza Modares [b], Alireza Alfi [a,*], Mohammad-Bagher Naghibi Sistani [b]

[a] Faculty of Electrical and Robotic Engineering, Shahrood University of Technology, Shahrood 36199-95161, Iran
[b] Department of Electrical Engineering, Ferdowsi University of Mashhad, Mashhad 91775-1111, Iran

ARTICLE INFO

Article history:
Received 13 July 2009
Received in revised form
10 April 2010
Accepted 4 May 2010
Available online 26 May 2010

Keywords:
Bilinear systems
Parameter estimation
Adaptive particle swarm optimization
Inertia weight
Optimization algorithms

ABSTRACT

Bilinear models can approximate a large class of nonlinear systems adequately and usually with considerable parsimony in the number of coefficients required. This paper presents the application of Particle Swarm Optimization (PSO) algorithm to solve both offline and online parameter estimation problem for bilinear systems. First, an Adaptive Particle Swarm Optimization (APSO) is proposed to increase the convergence speed and accuracy of the basic particle swarm optimization to save tremendous computation time. An illustrative example for the modeling of bilinear systems is provided to confirm the validity, as compared with the Genetic Algorithm (GA), Linearly Decreasing Inertia Weight PSO (LDW-PSO), Nonlinear Inertia Weight PSO (NDW-PSO) and Dynamic Inertia Weight PSO (DIW-PSO) in terms of parameter accuracy and convergence speed. Second, APSO is also improved to detect and determine varying parameters. In this case, a sentry particle is introduced to detect any changes in system parameters. Simulation results confirm that the proposed algorithm is a good promising particle swarm optimization algorithm for online parameter estimation.

1. Introduction

Bilinear model has been intensively studied in the literature because many objects in engineering, economics, ecology and biology etc. can be described by using a bilinear system (Lee et al., 1997; Mohler, 1991; Hua, 1990). A bilinear system is the simplest and most similar nonlinear system to a linear system in the form. It has many advantages, such as its special variable structure property which makes it advantageous on system modeling. The research of parameter estimation of the bilinear system has been carried out and received extensive attentions. Accurate knowledge of these parameters is important to form the control laws. Hence, it is of our interest to investigate an efficient model parameter tracking approach to achieve precise modeling results under different conditions without using complicated model structures.

Nowadays, a wide range of analytical techniques such as Recursive Least Square (RLS) (Godfrey and Jones, 1986), Recursive method of Instrumental Variable (RIV), Correlative Function method (COR), etc. exists for parameter estimation. RLS is usually considered that it is simple to use and costs little computation but it gives large estimation error when the system is influenced by colorful noises. Although COR can acquire more accurate parameter estimation but the estimation error is still not satisfying. Moreover, RIV can get accurate parameter but it costs large amount of complex computation. Most of these techniques have some fundamental problems including their dependence on unrealistic assumptions such as unimodal performance landscapes and differentiability of the performance function, and trapping in local minima (Ursem and Vadstrup, 2004). Also, if the searching space is undifferentiable or parameter span is nonlinear, traditional recurrent methods cannot gain the global optimization (Evsukoff et al., 2004; Montiel et al., 2004; Montiel et al., 2003; Juang et al., 2003; Goldkberg, 1989).

Heuristic algorithms especially with stochastic search techniques seem to be a more hopeful approach and provide a powerful means to solve this problem. These algorithms seem to be a promising alternative to traditional techniques, since they do not rely on any assumptions such as differentiability or continuity. In fact heuristic algorithms depend only on the objective function to guide the search. Because of this, Genetic Algorithm (GA) was used to estimate parameters for nonlinear systems (Kömürcü, 2008; Wang and Gu, 2007; Chang, 2007; Dai et al., 2002; Beligiannis et al., 2005). Although the GA is efficient to find the global minimum of the search space, it consumes too much search time which is not proper for online identification. The Particle Swarm Optimization (PSO) algorithm is an alternative. The main advantages of PSO are the simple concept, easy implementation and quick convergence. Due to this, recently PSO has attracted much attention and wide applications in various fields (Chang and Ko, 2009; Lin et al., 2008). The PSO algorithm is motivated by the behavior of organisms, such as fish schooling and bird flocking.

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

0952-1976/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2010.05.003

It is characterized as a simple concept, which is both easy to implement and computationally efficient. Unlike other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities (Abido, 2002). Based on this, this paper presents a novel PSO, namely Adaptive Particle Swarm Optimization (APSO), by introducing an adaptive inertia weight to rationally balance the global exploration and local exploitation abilities for basic PSO. An illustrative example for the modeling of bilinear systems is provided to confirm the validity, as compared with the GA and Linearly Decreasing Inertia Weight PSO (LDW-PSO), Nonlinear Inertia Weight PSO (NDW-PSO) and Dynamic Inertia Weight PSO (DIW-PSO) in terms of parameter accuracy and convergence speed. APSO is also improved to detect and determine the variation of parameters. In this case, a sentry particle is introduced to detect any changes in system parameters. If any change in parameters occurs, the sentry alerts the swarm to reset their best location memories and then the algorithm runs further to find the new optimum values. The performance of the proposed algorithm is demonstrated through identifying the parameters of a time varying bilinear system. Simulation results show that the proposed algorithm is a good promising particle swarm optimization algorithm for online parameter estimation.

The rest of paper is organized as follows: Next section describes a general form of problem formulation. Section 3 introduces the proposed APSO. In Section 4, the implementation of the proposed algorithm in online system parameter estimation is introduced. Sections 5 and 6 contain simulation results and conclusions, respectively.

2. Problem formulation

This paper considers the discrete bilinear system parameter estimation. The relationship between the input sequence $\{u(n)\}$and output sequence $\{y(n)\}$of such system is given by

$$y(n)=\alpha(n)+\sum_{i=1}^{p}a_i(n)y(n-i)+\sum_{i=1}^{p}b_i(n)u(n-i)+\sum_{i=1}^{p}\sum_{j=1}^{p}c_{ij}(n)u(n-i)y(n-j) \quad (1)$$

where p is the system order and $\sum_{i=1}^{p}\sum_{j=1}^{p}c_{ij}(n)u(n-i)y(n-j)$ is a bilinear term. Parameters which are needed to be estimated are a_i, b_i and c_{ij}.

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. Fig. 1 shows that the excitation input is given to both the real and the estimated systems. Then, the outputs are given as inputs to the fitness evaluator, where the fitness will be calculated. The sum of squared error for a number of samples is considered as "fitness of estimated model" defined by

$$\mathrm{SSE}=\sum_{k=1}^{N}e^2=\sum_{k=1}^{N}(y(k)-\hat{y}(k))^2 \quad (2)$$

where $y(k)$ and $\hat{y}(k)$ are real and estimated values in each sample, respectively and N is the number of given samples. The calculated fitness is then input to the identifier algorithm to identify the best parameters for estimated system in fitting procedure by minimizing the sum of squared errors in response to excitation input.

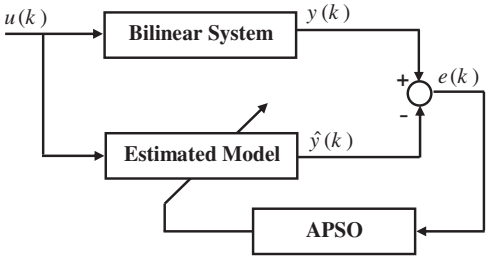

Fig. 1. The estimation process.

3. The proposed APSO

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 the beginning, a population with the size of particles is created. Then, the velocity of every particle is constantly adjusted according to corresponding particle's experience and particle's companions' experiences. It is expected that the particles will move towards better solution areas. The fitness of every particle can be evaluated according to the objective function of optimization problem. At each iteration, the velocity of every particle will be calculated as follows:

$$v_i^{t+1}=\omega\, v_i^t+c_1r_1(pbest_i^t-x_i^t)+c_2r_2(gbest^t-x_i^t) \quad (3)$$

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

$$x_i^{t+1}=x_i^t+v_i^{t+1} \quad (4)$$

The PSO algorithm performs repeated applications of the update equations above until the pre-specified number of generations G is reached.

Although PSO has shown some important advances by providing high speed of convergence in specific problems, it does exhibit some shortages. It found that PSO has a poor ability to search at a fine grain because it lacks velocity control mechanism (Angeline, 1998). Many approaches are attempted to improve the performance of PSO by variable inertia weight. The inertia weight is critical for the performance of PSO, which balances global exploration and local exploitation abilities of the swarm. A big inertia weight facilitates exploration, but it makes the particle long time to converge. Conversely, a small inertia weight makes the particle fast converge, but it sometimes leads to local optimal. Hence the linearly and nonlinearly decreasing inertia weight are proposed in the literature (Chang and Ko, 2009; Jiao et al., 2008; Yang et al., 2007; Chatterjee et al., 2006; Kennedy et al., 2001; Ratnaweera et al., 2004; Shi and Eberhart, 1998a,b). In the following, the three well-known mechanisms from above mentioned PSO algorithms are described that are used for comparison with the proposed PSO in the problem in hand. In the first one namely LDW-PSO, the inertia weight is adapted linearly as follows (Shi and Eberhart, 1998a,b):

$$\omega^t=\omega_{\min}+\frac{iter_{\max}-t}{iter_{\max}}.(\omega_{\max}-\omega_{\min}) \quad (5)$$

where $iter_{\max}$ is the maximal number of iterations, t is the current number of iterations. So as iterations go, ω decreases linearly from $\omega_{\max}$ to $\omega_{\min}$. In the second one namely NDW-PSO, the inertia weight is adapted nonlinearly as follows (Chatterjee et al., 2006):

$$\omega^t = \omega_{\min} + (\frac{iter_{\max} - iter}{iter_{\max}})^n . (\omega_{\max} - \omega_{\min}) \quad (6)$$

where n is the nonlinear modulation index. In the last one namely DIW-PSO, the inertia weight is adapted dynamically as follows (Jiao et al.; 2008):

$$\omega^t = \omega_{init} \times u^{-t} \quad (7)$$

where $\omega_{init} \in (0,1]$ is the initial inertia weight and $u \in [1.0001, 1.005]$.

Nevertheless these algorithms improve the performance of PSO, they cannot truly reflect the actual search process without any feedback taken from how far particle's fitness are from the estimated (or real) optimal value, when the real optimal value is known in advance. Actually, for 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 larger values. Conversely, only a small movement is needed and so inertia weight must set to a small value to facilitate finer local explorations. Furthermore, 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. In fact, during the search every particle dynamically changes its position, so every particle locates in a complex environment and faces different situation. Therefore, every particle may have different trade off between global and local search abilities.

Motivated by the aforementioned, in this paper, the inertia weight is dynamically adapted for every particle by considering a measure called Adjacency Index (AI), which characterizes the nearness of individual fitness to the real optimal solution. Based on this index, every particle could decide how to adjust the values of inertia weight. For this purpose, the velocity updating rules in the proposed APSO is given by

$$v_i^{t+1} = \omega_i^t v_i^t + c_{1i}^t r_1 (pbest_i^t - x_i^t) + c_2^t r_2 (gbest^t - x_i^t) \quad (8)$$

Compared with that in PSO, the velocity updating Eq. (3) has two different characteristics:

(1) To incorporate the difference between particles into PSO, so that it can simulate a more precise biological model, the inertia weight is variable with the number of particles.
(2) To truly reflect the actual search process, the inertia weight is set according to feedback taken from particles best memories.

Definition 1. The Adjacency Index (AI) for every particle in tth iteration is defined as follows:

$$AI_i^t = \frac{F(pbest_i^1) - F_{KN}}{F(pbest_i^t) - F_{KN}} - 1 \quad (9)$$

where $F(pbest_i^t)$ is the fitness of the best previous position of ith particle and F_{KN} is the known real optimal solution value.

In the first iteration, AI_i is zero for ith particle and finally if $F(pbest_i) = F_{KN}$ then AI_i is infinite. A small AI_i means that the fitness of ith particle is far away from the real optimal value and it needs a strong global exploration therefore, a large inertia weight. On the other hand, a big AI_i means that ith particle has a high adjacency to the real optimum and so it needs a strong local exploitation, therefore a small inertia weight. Hence, the value of inertia weight for every particle in tth iteration is dynamically calculated with the following transform function.

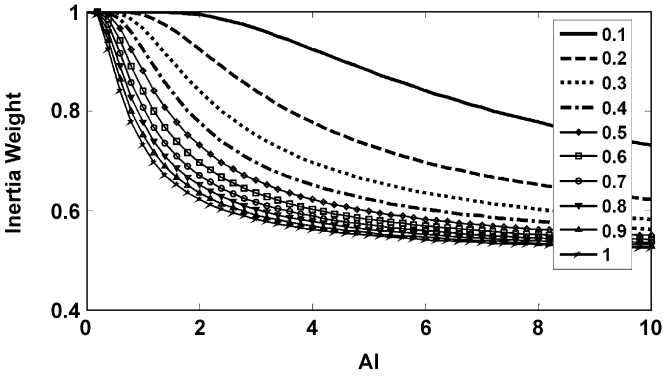

Fig. 2. Inertia weight versus AI with different values of α.

Remark 1. Note that in the most of engineering optimization problem such as system identification considered in this paper, the optimal value F_{KN} is known in advance. However, if the optimal value is unknown, the Eq. (9) can be modified as

$$AI_i^t = \frac{F(pbest_i^1)}{F(pbest_i^t)} - 1 \quad (10)$$

In this case, for each particle the Adjacency Index changes according to the rate of its personal best fitness improvement.

Definition 2. The transfer function is

$$\omega_i^t = \frac{1}{1 + e^{-(\alpha \times AI_i^t)^{-1}}} \quad (11)$$

where α is a positive constant in the range (0,1]. Under the assumption and definitions above, it can be concluded that $0.5 \le \omega_i < 1$.

Fig. 2 shows the change of inertia weight ω with respect to AI with different values of α. The parameter α controls the decreasing speed of inertia weight. In order to observe the impact of α on the performance of APSO, parameter α is varied from 0.1 to 1 with step size 0.1.

According to Eqs. (9) and (11), during the search, the particles face different finesses; as a result they get different values of AI and then inertia weight. While the fitness of a particle is far away from the real global optimal, AI for this particle has a small value (a low adjacency) and the value of inertia weight will be large resulting strong global search abilities and locate the promising search areas. Meanwhile, the fitness of a particle achieves near the real global optimal, AI for this particle has a big value (a high adjacency) and inertia weight will be set small, depending on the nearness of its best fitness to the optimal value, to facilitate a finer local explorations and so accelerate convergence.

4. Implementation of APSO

In this section, the procedure of APSO in online system parameter identification is described. In this case, each particle represents all parameters of estimated model. The proposed algorithm sequentially gives a data set by sampling periodically. While starting, in the first period, the best system parameter is found by minimizing the SSE introduced in Eq. (2). In this case, the simulation for next period does not begin until the fitness of global best becomes lower than a predefined threshold. After that, the estimated parameters will not be updated unless a change in the system parameters is detected. In order to detect any change

in system parameters, the global optimum in the later period is noticed as a sentry particle. In the beginning of each of the next periods, the sentry reevaluates its fitness and if the fitness changes significantly or it becomes bigger than a predefined threshold, the changes in parameters are confirmed. If no changes are detected, the algorithm leaves this period without changing the positions of particles. In contrast, when any change in parameters occurs, the sentry alerts the swarm to reset their best location memories and then the algorithm runs further to find the new optimum values. For this purpose, the fitness of global optimum particle and personal bests of all particles are evaporated at the rate of a big evaporation constant. As a result, other particles have a chance to find better solutions than those stored on their pervious global and personal memories. Moreover, the velocities of particles are increased to search in a bigger solution space for new optimal solution.

Generally speaking, when a change in system parameters is detected the following changes in algorithm are done and then the proposed algorithm runs to identify the new optimal parameters:

$$fitness(pbest_i) = fitness(pbest_i) \times T \quad i = 1,...,S \tag{12}$$

$$fitness(gbest) = fitness(gbest) \times T \tag{13}$$

$$V = V + \beta V_{\max} \tag{14}$$

where T is a big evaporation constant and β is a random variable between 0 and 1. The procedure for this algorithm is summarized as follows:

Step 1. Give a data set in the current period.

Step 2. If it is the first period, initialize positions and velocities of a group of particles, then go to Step 4.

Step 3. Evaluate the fitness of the sentry particle. If any change in system parameters is detected by the sentry particle, reset the best location memories and velocities of particles using Eqs. (11)–(14). Else go to Step 1.

Step 4. Evaluate fitness of each particle using Eq. (2).

Step 5. If the new position of ith particle is better than $Pbest_i$, set $Pbest_i$ as the new position of the ith particle. If the fitness of best position of all new particles is better than fitness of *gbest*, then *gbest*is updated and stored.

Step 6. Calculate the inertia weight using Eq. (11).

Step 7. Update the position and velocity of each particle according to the Eqs. (4) and (8).

Step 8. If the global optimum fitness is lower than a redefined threshold, output the global optimum and label it as the sentry particle, then go to the step 1. Else go to Step 4.

5. Simulation results

This section demonstrates the feasibility of the APSO-based parameter system identification. The results are compared to those obtained by LDW-PSO, NDW-PSO, DIW-PSO and GA. In all PSO algorithms, $c_1 = c_2 = 2$ (Kennedy and Eberhart, 1995). In LDW-PSO and NDW-PSO, ω decreases from 0.9 to 0.4. Moreover, in NDW-PSO n is set to 1.2 (Chatterjee et al., 2006). In DIW-PSO, u is set to 1.0002 (Jiao et al., 2008). In APSO, the parameter ω is determined using Eq. (11). In addition, in GA, the crossover probability P_c and the mutation probability P_m are set to 0.8 and 0.1, respectively. To perform fair comparison, the same computational effort is used in GA, LDW-PSO, NDW-PSO, DIW-PSO and APSO. That is, the maximum generation, population size and searching range of the parameters in GA are the same as those in LDW-PSO, NDW-PSO, DIW-PSO and APSO.

Table 1
The mean best fitness values with different values of α.

α	SSE	
	Example 1	Example 2
0.1	2.53×10^{-17}	4.22×10^{-16}
0.2	3.98×10^{-21}	7.26×10^{-17}
0.3	4.12×10^{-21}	3.19×10^{-18}
0.4	3.07×10^{-21}	2.89×10^{-19}
0.5	8.02×10^{-22}	2.12×10^{-20}
0.6	7.12×10^{-22}	5.52×10^{-18}
0.7	1.22×10^{-19}	1.21×10^{-15}
0.8	4.41×10^{-21}	5.02×10^{-17}
0.9	5.79×10^{-18}	7.87×10^{-15}
1	8.85×10^{-18}	7.02×10^{-14}

In order to observe the impact of α on the performance of APSO, different values of α, 20 particles, 200 and 400 maximum iterations for Examples 1 and 2, respectively, were conducted on parameter estimation of the following examples. For each experimental setting, 20 runs of the algorithm were performed. Table 1 listed the mean best fitness values averaged over 20 runs. It is clear that the values in range [0.2, 0.8] for α can all lead to acceptable performance. In present paper, α is set to 0.5.

Simulation results have been carried out in two cases. In the first case, the proposed algorithm has been compared with GA, LDW-PSO, NDW-PSO and DIW-PSO in offline parameter identification in terms of convergence speed and accuracy using two examples described by (Wang and Gu, 2007). In the second case, the proposed APSO is applied to online parameter identification of Example 1 for its performance validation.

Example 1. A bilinear system is given as follows:

$$y(k+1) = ay(k-1) + bu(k) - cy(k)u(k) \tag{15}$$

where $a = 0.898$, $b = 0.28$ and $c = -0.106$.

Example 2. A MIMO bilinear system was given as follows:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + B \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} u_1(k) + C \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} u_2(k) + D \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix} \tag{16}$$

where

$$A = \begin{bmatrix} -0.2 & 0.25 \\ 0.1 & -0.18 \end{bmatrix},\ B = \begin{bmatrix} 0.61 & 0.1 \\ 0.31 & 0.35 \end{bmatrix},\ C = \begin{bmatrix} -0.31 & 0.0 \\ 0.5 & 0.0 \end{bmatrix},$$
$$D = \begin{bmatrix} -0.02 & 0.5 \\ 0.53 & -0.4 \end{bmatrix}.$$

5.1. Case 1: Offline

In these examples, for the APSO, the known optimal value F_{KN} is zero. In Example 1, the optimization process is repeated 20 times independently. The average and Standard deviation (denoted by Std) of results using GA (Wang and Gu, 2007), LDW-PSO, NDW-PSO, DIW-PSO and APSO are listed in Table 2. Figs. 3–5 depict the great success of optimization process by using APSO in compared with DIW-PSO for the identified parameters a, b and c, respectively. The aim is to avoid confusion between the results of APSO and other algorithms, so only the result of DIW-PSO which is the best result obtained among other algorithms in terms of convergence speed are compared, excluding the rest. Moreover, the convergence of the optimal SSE at each generation is plotted for all algorithms in Fig. 6. Simulations results confirm the superiority of APSO algorithm in terms of accuracy and convergence speed without the premature convergence problem.

Table 2
Parameter estimation of Example 1 by using GA (Wang and Gu, 2007), LDW-PSO, NDW-PSO, DIW-PSO and APSO algorithms.

	a	*b*	*c*	SSE	Std
Real Value	0.898	−0.106	0.284	–	–
GA	0.897268	−0.107984	0.284729	5.2×10^{-3}	0.08763
LDW-PSO	0.89800004	−0.10600028	0.28400002	8.7×10^{-11}	9.1×10^{-9}
NDW-PSO	0.89800003	−0.10600022	0.28399992	2.3×10^{-13}	3.7×10^{-11}
DIW-PSO	0.89800001	−0.10600009	0.28399995	1.1×10^{-14}	7.7×10^{-13}
APSO	**0.89800000**	**−0.10600000**	**0.28399999**	$\mathbf{5.2\times10^{-22}}$	$\mathbf{1.4\times10^{-21}}$

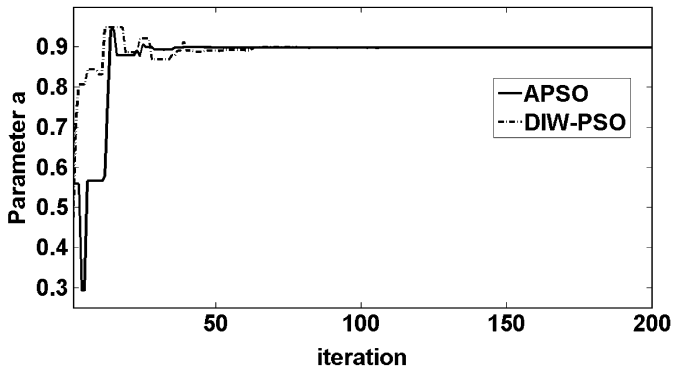

Fig. 3. Comparison of trajectories of parameter *a*.

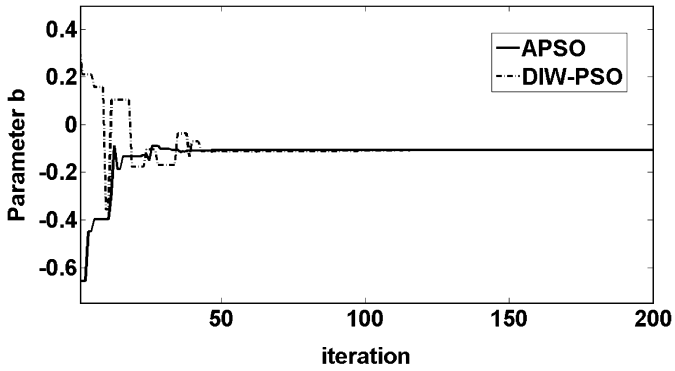

Fig. 4. Comparison of trajectories of parameter *b*.

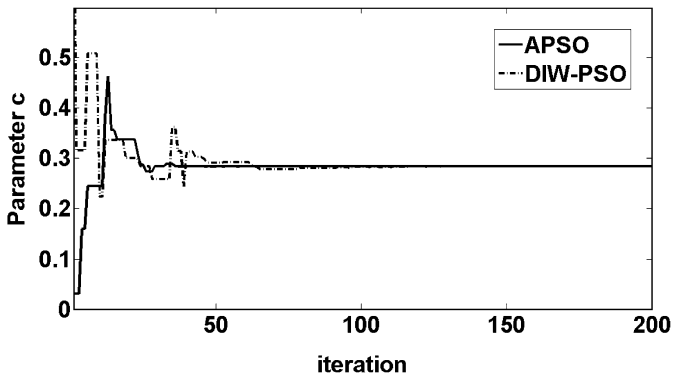

Fig. 5. Comparison of trajectories of parameter *c*.

In Example 2, the average and Std of results using GA (Wang and Gu, 2007), LDW-PSO, NDW-PSO, DIW-PSO and APSO are listed in Tables 3 and 4. It is obvious that the results of the proposed APSO are good promising as compared with other algorithms. In addition, Figs. 7–10 illustrate three instance system parameters including A(1,1), A(1,2), B(1,1) obtained by APSO and DIW-PSO, respectively and the convergence trajectories of the optimal SSE at each generation for all algorithms. It again confirms the superiority of APSO algorithm in terms of convergence speed without the premature convergence problem.

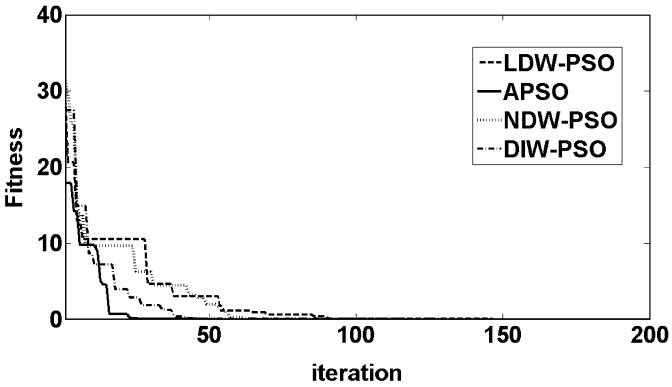

Fig. 6. Comparison of convergence of objective function for Example 1.

5.2. Case 2: Online

In this case, the variation of system parameters is considered. This study is necessary to justify tracking the changes of system parameters using the proposed algorithm. If a change in the model parameter is detected by sentry particle, the APSO continues to run. In this moment, the simulation for next period does not begin until the fitness of global best becomes lower than a threshold of 10^{-5} in this period. After finding optimal parameters in this period, there will be no APSO iteration unless another change detects in system parameter. Based on this, a sudden change is applied for the system parameters. Notice that the slower parameters of time-varying change, the smaller choice threshold is to have better result.

The separated parts (I), (II) and (III) in Figs. 11–13 illustrate how the proposed algorithm tracks variation of system parameters in Example 1. In part (I), the original parameters which are used in offline section are given. In part (II), parameter *a* is changed from 0.898 to 1, while *b* and *c* are kept unchanged. Finally, in part (III), parameter *a* is kept unchanged whereas parameters *b* and *c* are simultaneously varied from −0.106 to −0.12 and from 0.284 to 0.3, respectively. Considering Figs. 5–7, it is obvious that the proposed algorithm can track any change in parameters. The dashed lines in these figures signify the moment that the sentry particle has detected some change in system parameters. Moreover, it can be seen that unchanged parameters, *b* and *c* in part (II) and *a* in part (III) reach on the previous values when simulation is terminated at the end of each part. Table 5 illustrates the results obtained by APSO in online identification. The results indicate that APSO successfully identify parameters variations.

Table 3
Parameter estimation of Example 2 by using GA (Wang and Gu, 2007), LDW-PSO, NDW-PSO, DIW-PSO and APSO.

	Real value	GA	LDW-PSO	NDW-PSO	DIW-PSO	**APSO**
A(1,1)	−0.2	−0.2041	−0.199999978950624	−0.19999999822718	−0.200000003962381	**−0.20000000000013**
A(1,2)	0.25	0.2307	0.250000010999967	0.25000000089117	0.249999996860713	**0.25000000000004**
A(2,1)	0.1	0.0876	0.099999992025016	0.09999999911527	0.100000001450311	**0.10000000000021**
A(2,2)	−0.18	−0.1650	−0.180000011561963	−0.18000000010448	−0.179999999177745	**−0.18000000000023**
B(1,1)	0.61	0.5925	0.610000036318299	0.61000000300658	0.610000000085002	**0.61000000000054**
B(1,2)	0.1	0.0832	0.099999991935472	0.10000000026139	0.099999998392734	**0.10000000000009**
B(2,1)	0.31	0.3263	0.309999982789969	0.30999999801552	0.310000001289933	**0.31000000000002**
B(2,2)	0.35	0.3584	0.349999977491103	0.34999999784326	0.349999999352850	**0.35000000000046**
C(1,1)	−0.31	−0.3075	−0.309999976984384	−0.30999999555033	−0.309999998427939	**−0.31000000000081**
C(1,2)	0.0	−0.0096	−0.000000038392281	0.00000000306839	−0.000000002956280	**−0.00000000000075**
C(2,1)	0.5	0.4887	0.499999981178399	0.49999999599767	0.499999999648942	**0.50000000000000**
C(2,2)	0.0	0.0104	0.000000021451939	0.00000000505569	0.000000000152011	**0.00000000000090**
D(1,1)	0.02	0.0469	0.020000022378077	0.02000000131165	0.020000001340292	**0.02000000000024**
D(1,2)	0.5	0.5155	0.499999979634539	0.50000000028094	0.499999998357644	**0.50000000000012**
D(2,1)	0.53	0.5202	0.529999998257093	0.53000000026492	0.530000001308541	**0.53000000000000**
D(2,2)	−0.4	−0.3978	−0.400000013756547	−0.40000000101925	−0.399999999755147	**−0.40000000000001**

Table 4
SSE and Std obtained by using GA (Wang and Gu, 2007), LDW-PSO, NDW-PSO, DIW-PSO and APSO algorithms for Example 2.

	GA	LDW-PSO	NDW-PSO	DIW-PSO	**APSO**
SSE	3.1×10^{-3}	1.8×10^{-9}	1.3×10^{-12}	4.1×10^{-14}	$\mathbf{9.7\times10^{-19}}$
Std	0.0623	4.4×10^{-6}	4.9×10^{-9}	2.8×10^{-12}	$\mathbf{7.4\times10^{-18}}$

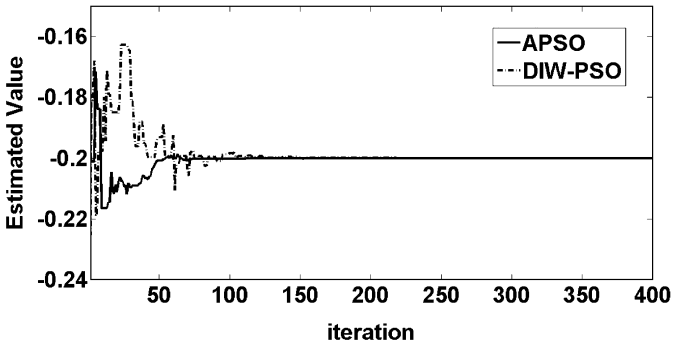

Fig. 7. Comparison of trajectories of parameter A(1,1).

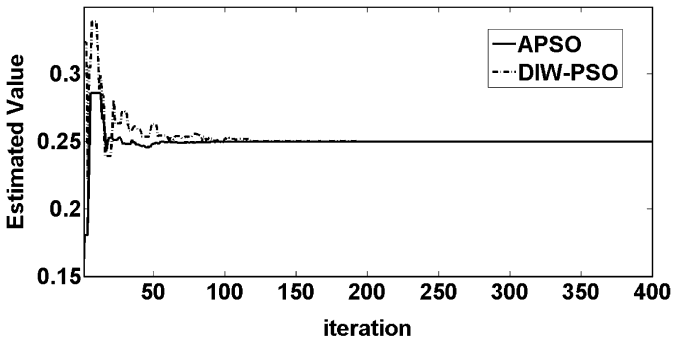

Fig. 8. Comparison of trajectories of parameter A(1,2).

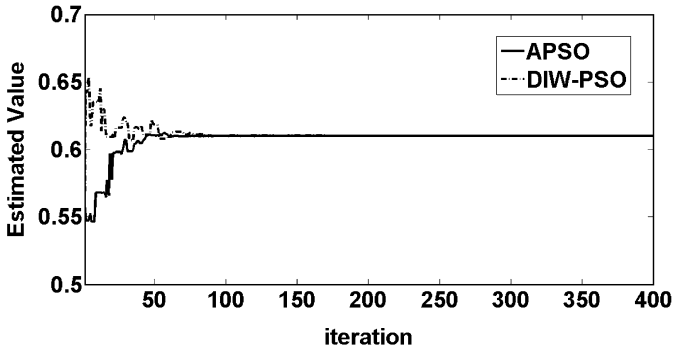

Fig. 9. Comparison of trajectories of parameter B(1,1).

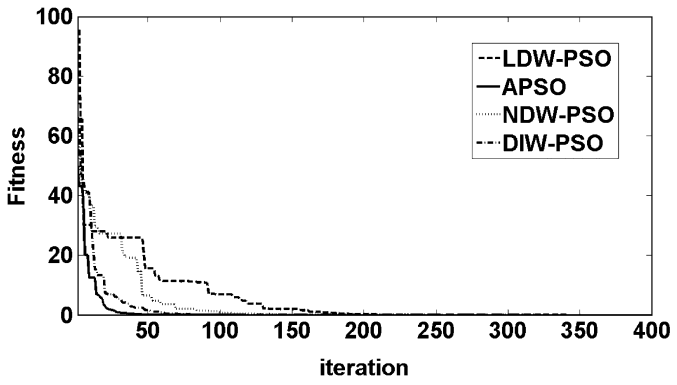

Fig. 10. Comparison of convergence of objective function for Example 2.

6. Conclusion

Online estimation of systems with unknown and varying parameter is a challenging problem. The difficulties of online implementation mainly come from the unavoidable computational time to find a solution. This paper presented a novel nature-inspired optimization technique, namely APSO to solve this problem. An adaptive inertia weight mechanism was proposed in APSO to increase the convergence speed and accuracy of the basic PSO to save tremendous computation time. APSO was also improved to detect and determine varying parameters. Although the feasibility of APSO was shown for both offline and online identification of bilinear systems, it does not have any restriction to other systems. The future work is to apply APSO for both system identification and control in an adaptive manner.

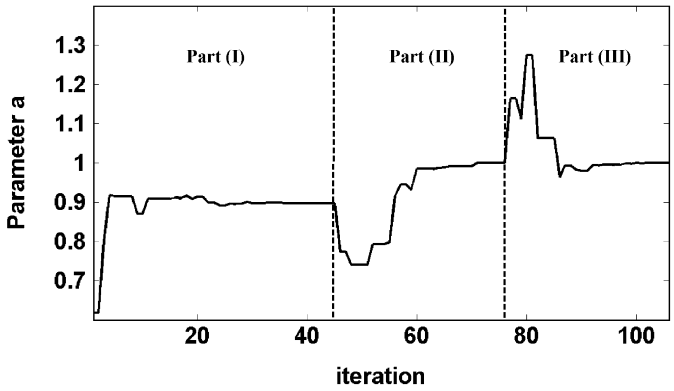

Fig. 11. APSO process for tracking variable parameter *a*.

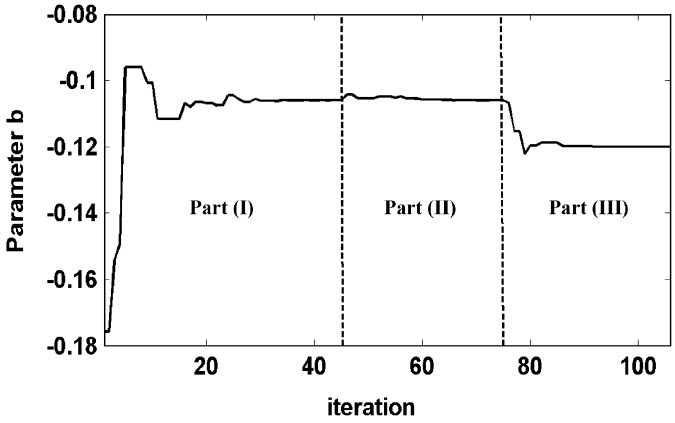

Fig. 12. APSO process for tracking variable parameter *b*.

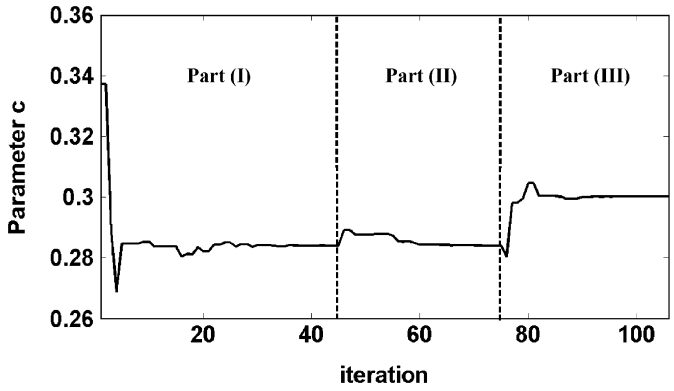

Fig. 13. APSO process for tracking variable parameter *c*.

Table 5
Online estimation results of Example 1 by APSO.

	Part (I)		Part (II)		Part(III)	
	Actual	Predicted	Actual	Predicted	Actual	Predicted
a	0.89800	**0.89800**	1.00000	**1.00003**	1.00000	**1.00003**
b	−0.10600	**−0.10607**	−0.10600	**−0.10601**	−0.12000	**−0.12001**
c	0.28400	**0.28401**	0.28400	**0.28399**	0.30000	**0.30003**

References

Abido, M.A., 2002. Optimal design of power-system stabilizers using particle swarm optimization. IEEE Transactions on Energy Conversion 17 (3), 406–413.

Angeline, P.J., 1998. Evolutionary optimization versus particle swarm optimization: philosophy and performance differences, Lecture Notes in Computer Science 1447. Springer-Verlag, Berlin Heidelberg 601 – 610.

Beligiannis, G., Skarlas, L., Likothanassis, S., Perdikouri, K., 2005. Nonlinear model structure identification of complex biomedical data using a genetic-programming-based technique. IEEE Transactions on Instrumentation and Measurement 54 (6), 2184–2190.

Chang, W.D., 2007. Nonlinear system identification and control using a real-coded genetic algorithm. Applied Mathematical Modelling 31, 541–550.

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 Application 36 (3), 6809–6816.

Chatterjee, A., Siarry, P., 2006. Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization. Computers and Operations Research 33 (3), 859–871.

Dai, D., Ma, X.K., Li, F.C., You, Y., 2002. An approach of parameter estimation for a chaotic system based on genetic algorithm. Acta Physica Sinica 11, 2459–2462.

Evsukoff, A., Ebecken, N., 2004. Identification of recurrent fuzzy systems with genetic algorithms. In: Proceedings of the IEEE International Conference on Fuzzy Systems, pp. 1703–1708.

Godfrey, K., Jones, P., 1986. Signal Processing for Control. Springer-Verlag, Berlin.

Goldkberg, D., 1989. Genetic algorithms in search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Hua, X., 1990. Modeling and Control of Bilinear System. Northeast Chemical Engineering Institute Press.

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

Juang, J.G., 2003. Application of genetic algorithm and recurrent network to nonlinear system identification. In: Proceedings of the IEEE Conference on Control Applications, pp. 129–134.

Kennedy, J., Eberhart, R.C., 1995. Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948.

Kennedy, J., Eberhart, R.C., Shi, Y., 2001. Swarm intelligence. Morgan Kaufmann Publishers, San Francisco.

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

Lee, S.H., Kong, J.S., Seo, J.H., 1997. Observers for bilinear systems with unknown inputs and application to superheater temperature control. Control Engineering Practice 5 (4), 493–506.

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.

Mohler, R.R., 1991. Nonlinear Systems Application to Bilinear Control. Prentice-Hall.

Montiel, O., Castillo, O., Melin, P., Diaz, A.R., Sepulveda, R., 2004. Evolutionary nonlinear system identification. In: Proceedings of the International Conference on Information Acquisition, pp. 98–104.

Montiel, O., Castillo, O., Melin, P., Diaz, A.R., Sepulveda, R., 2003. The evolutionary learning rule for system identification. Applied Soft Computing 3 (4), 343–352.

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

Shi, Y., Eberhart, R.C., 1998a. 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., 1998b. A modified particle swarm optimizer. In: Proceedings of the Conference on Evolutionary Computation, pp. 69–73.

Ursem, R.K., Vadstrup, P., 2004. Parameter identification of induction motors using stochastic optimization algorithms. Applied Soft Computing 4 (1), 49–64.

Wang, Z., Gu, H., 2007. Parameter identification of bilinear system based on genetic algorithm, Lecture Notes in Computer Science 4688. Springer-Verlag, Berlin Heidelberg 83 – 91.

Yang, X., Yuan, J., Yuan, J., Mao, H., 2007. A modified particle swarm optimizer with dynamic adaptation. Applied Mathematics and Computation 189 (2), 1205–1213.