Seeker optimization algorithm for parameter estimation of time-delay chaotic systems

Chaohua Dai,[1,*] Weirong Chen,[1] Lixiang Li,[2] Yunfang Zhu,[3] and Yixian Yang[2]

[1]*The School of Electrical Engineering, Southwest Jiaotong University, Chengdu 610031, China*

[2]*Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, P.O. Box 145, Beijing 100876, China*

[3]*Department of Computer and Communication Engineering, E'mei Campus, Southwest Jiaotong University, E'mei 614202, China*

(Received 12 April 2010; revised manuscript received 23 October 2010; published 14 March 2011)

Time-delay chaotic systems have some very interesting properties, and their parameter estimation has received increasing interest in the recent years. It is well known that parameter estimation of a chaotic system is a nonlinear, multivariable, and multimodal optimization problem for which global optimization techniques are required in order to avoid local minima. In this work, a seeker-optimization-algorithm (SOA)-based method is proposed to address this issue. In the SOA, search direction is based on the empirical gradients by evaluating the response to the position changes, and step length is based on uncertainty reasoning by using a simple fuzzy rule. The performance of the algorithm is evaluated on two typical test systems. Moreover, two state-of-the-art algorithms (i.e., particle swarm optimization and differential evolution) are also considered for comparison. The simulation results show that the proposed algorithm is better than or at least as good as the other two algorithms and can effectively solve the parameter estimation problem of time-delay chaotic systems.

DOI: 10.1103/PhysRevE.83.036203 PACS number(s): 05.45.Ac, 05.45.Gg, 05.45.Pq

I. INTRODUCTION

The problem of parameter estimation in chaotic systems is treated as an important issue in nonlinear science and has attracted ever-growing attention of researchers in the past decades [1–21]. Due to the complexity and the unstable dynamic behavior of chaotic systems, it is not easy to determine their parameters, and parameter estimation of chaotic systems is well known as a nonlinear, multivariable, and multimodal optimization problem for which global optimization techniques are required in order to avoid local minima [22–28].

In many real systems, such as chemical processes, biological systems, rolling mill systems, and economic systems, time delay is commonly encountered. Since Mackay *et al.* [29] first found chaos in a time-delay system, control and synchronization of time-delay chaotic systems has become an active area of research [27,28,30–35]. Time-delay chaotic systems can produce chaotic attractors with an arbitrarily large number of positive Lyapunov exponents, which is a very interesting property because it can improve the level of security in secure communication. To achieve the goal of control and synchronization of time-delay chaotic system, the parameters and time-delay constant should be estimated when they are unknown [27,28]. Because the conventional gradient-based methods may easily get stuck in local minima, recently some intelligent optimization techniques have been proposed for the application of parameter estimation of time-delay chaotic systems, including particle swarm optimization (PSO) [27] and differential evolution (DE) [28]. Although DE and PSO have received much interest from the evolutionary computation (EC) community, it does not mean that there are not any limitations to them. Several studies have shown that the DE is prone to not only premature convergence but also stagnation [36] and that a successful location of the global optimum depends on choosing the correct control parameters [37]. Meanwhile, the performance of PSO also depends on its parameters [38] and may be influenced by premature convergence and stagnation problems [36,39].

Human beings are the highest-ranking animals in nature. Optimization tasks are often encountered in many areas of human life [40], and the search for a solution to a problem is one of the basic behaviors to all mankind [41]. In the course of evolution, human beings have accumulated a great wealth of experience to optimally solve their practical problems. Why not learn from human beings (i.e., ourselves)? The seeker optimization algorithm (SOA) used in this paper is just such a paradigm, which takes optimization processes as a search by a human team for an optimal solution and focuses directly on human group searching behaviors to be simulated for real-parameter optimization. As a new population-based heuristic search algorithm, SOA, also called a human group optimizer (HGO), has been applied to function optimization [42], proton exchange membrane fuel cell model optimization [43], optimal reactive power dispatch [44,45], digital IIR filter design [46], etc. In our previous works [42–46], some concepts in SOA may be confusing and were left to be further elaborated. Moreover, Our previous works did not explicitly analyze the impact of the main parameters of the SOA on the optimization performance. In this paper, some concepts in SOA are further clarified, especially the relationship between population and subpopulation and the inter-subpopulation learning strategy. In addition, this paper presents the in-depth analysis on the impact of the main parameters of the SOA on the parameter-estimation performance. Then the SOA is applied to parameter estimation of time-delay chaotic systems on two typical time-delay chaotic systems. Unlike Refs. [27,28], where the PSO and DE were used for the same tasks but not compared with other methods, this paper presents a comparison of SOA with both PSO [27] and DE [28]. The simulation results show that SOA has better, or at least equally good, performance than the other algorithms and becomes a promising candidate for parameter estimation of time-delay chaotic systems.

*dchzyf@yahoo.com.cn

In the last years, the covariance matrix adaptation evolution strategy (CMA-ES) [47] has also attracted a certain amount of attention of researchers. The CMA-ES is a class of continuous evolutionary algorithm (EA) with the selection, recombination, and mutation operators. It generates new population members by sampling from a probability distribution that is constructed during the optimization process. On a contrary, the SOA is a class of swarm intelligence algorithms without the traditional evolutionary operators. Because the CMA-ES is a different algorithm from the SOA and has also not been applied to parameter estimation of time-delay chaotic systems, we do not consider it in the Sec. IV.

The rest of this paper is organized as follows: Section II presents the mathematical formulation of parameter estimation of time-delay chaotic systems. In Sec. III, SOA is described in detail. In Sec. IV, SOA is evaluated on two typical systems with comparisons of other algorithms. Then, in Sec. V, further analysis on the parameters of SOA is conducted. Finally, the conclusion is drawn in Sec. VI, clearly different from the CMA-ES.

II. PROBLEM FORMULATION

The problem formulation of parameter estimation for time-delay chaotic systems is presented in this section. Consider the following time-delay chaotic system:

$$\dot{\boldsymbol{Y}} = F[\boldsymbol{Y}(t),\boldsymbol{Y}(t-\tau),\boldsymbol{Y}_0,\boldsymbol{x}], \tag{1}$$

where $\boldsymbol{Y}(t) = [y_1(t),y_2(t),\ldots,y_n(t)]^{\mathrm{T}}$ is the n-dimensional state vector with the initial state $\boldsymbol{Y}_0$ and $\boldsymbol{x} = (x_1,x_2,\ldots,x_D)^{\mathrm{T}}$ denotes the D-dimensional parameter vector of the chaotic system.

For the purpose of parameter estimation, let the system structure be assigned, and the estimated system can be described as follows:

$$\dot{\tilde{\boldsymbol{Y}}} = F[\tilde{\boldsymbol{Y}}(t),\tilde{\boldsymbol{Y}}(t-\tilde{\tau}),\boldsymbol{Y}_0,\tilde{\boldsymbol{x}}], \tag{2}$$

where $\tilde{\boldsymbol{Y}}$ is the n-dimensional state vector and $\tilde{\boldsymbol{x}} = (\tilde{x}_1,\tilde{x}_2,\ldots,\tilde{x}_D)^{\mathrm{T}}$ is the estimated parameter vector. The time delay $\tilde{\tau}$ is treated as a component of the estimated parameter vector.

In this case, the parameters of the estimated system are successively adjusted by an optimization method until the error between the state vectors of the original system and the estimated system is minimized (shown in Fig. 1). In other words, the task is formulated as a minimization optimization problem [27,28]:

$$\min \boldsymbol{J}(\tilde{\boldsymbol{x}}) = \sum_{t=1}^{N} \| \boldsymbol{Y}(t) - \tilde{\boldsymbol{Y}}(t) \|^2 , \tag{3}$$

where N is the number of samples used for the calculation of objective function and $\boldsymbol{Y}(t)$ and $\tilde{\boldsymbol{Y}}(t)$ $(t = 1,2,\ldots,N)$ denote the state vectors of the original system and the estimated system at time t, respectively.

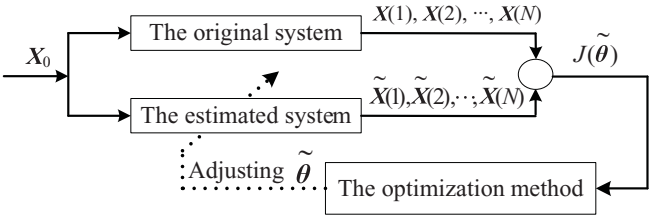


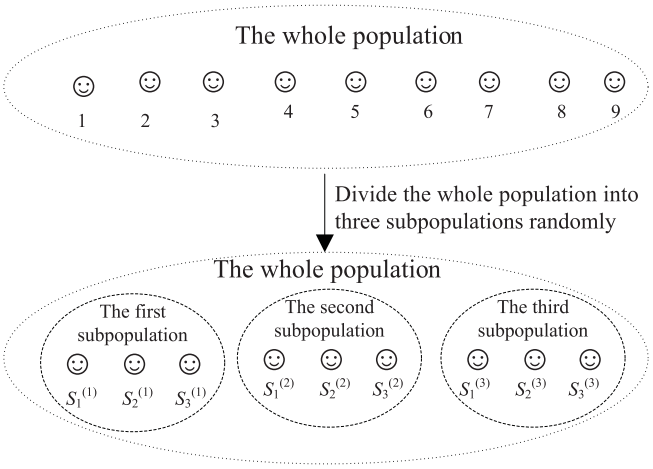
FIG. 1. Block diagram of the parameter-estimation process for time-delay chaotic systems.

III. SEEKER OPTIMIZATION ALGORITHM

In this section, the main material about the SOA from Refs. [42–46] is included in order to make the introduction of the SOA method more self-contained. SOA operates on a set of potential solutions called a search population (i.e., human group or swarm). The individual (i.e., a potential solution) of this population is called a seeker (i.e., person or agent). In order to add a social component for social sharing of information, a neighborhood is defined for each seeker. In the present simulations, for example, the population is randomly categorized into $K = 3$ subpopulations (every subpopulation has the same size), and all seekers in the same subpopulation constitute a neighborhood. A relationship chart representing population and subpopulations is also presented as Fig. 2.

A. Implementation of SOA

Assume that the optimization problems to be solved are minimization problems. The main steps of SOA are shown as Fig. 3. A search direction $\vec{d}_i(t) = [d_{i1},d_{i2},\ldots,d_{iD}]$ and a step length vector $\vec{\alpha}_i(t) = [\alpha_{i1},\alpha_{i2},\ldots,\alpha_{iD}]$ are computed (see Sec. III B and III C) for the ith seeker at time step t, where $\alpha_{ij}(t) \geqslant 0$, $d_{ij} \in \{-1,0,1\}$, $i = 1,2,\ldots,s$, $j = 1,2,\ldots,D$,



Note: 1. take the population size s=9 for example;
2. $S_i^{(k)}$ denotes the ith seeker of the kth subpopulation, i=1,2,3, k=1,2,3;
3. $S_1^{(1)}$, $S_2^{(1)}$, $S_3^{(1)}$, $S_1^{(2)}$, $S_2^{(2)}$, $S_3^{(2)}$, $S_1^{(3)}$, $S_2^{(3)}$, $S_3^{(3)}$ are the mutually exclusive integers within [1,9], i.e., $S_1^{(1)}$, $S_2^{(1)}$, $S_3^{(1)}$, $S_1^{(2)}$, $S_2^{(2)}$, $S_3^{(2)}$, $S_1^{(3)}$, $S_2^{(3)}$, $S_3^{(3)} \in [1,9]$ and $S_1^{(1)} \neq S_2^{(1)} \neq S_3^{(1)} \neq S_1^{(2)} \neq S_2^{(2)} \neq S_3^{(2)} \neq S_1^{(3)} \neq S_2^{(3)} \neq S_3^{(3)}$

FIG. 2. The relationship between population and subpopulation.

The generation $t \leftarrow 0$, the number of function evaluations $n \leftarrow 0$;
Generating s positions uniformly and randomly in search space;
Calculating the objective function values for all the seekers, and $n=s$;
Saving the minimum objective function value so far (i.e., the historical best value of the whole population in the algorithm) and its position;
Repeat
  $t \leftarrow t+1$;
  For $i=1$ to s
    Calculating $\vec{d}_i(t)$ by Eqs. (9) - (13);
    Calculating $\vec{\alpha}_i(t)$ by Eqs. (14) - (16);
    Updating the ith seeker's position by Eq. (4);
    Calculating the objective function value of the ith seeker;
    $n \leftarrow n+1$;
  End of for loop
  Saving the minimum objective function value so far and its position;
  Implementing the inter-subpopulation learning operation;
Until the t value equals the maximum generations (300 in this paper) or the minimum objective function value so far is equal to zero.
Outputting the minimum objective function value so far and its position (i.e., the final solution obtained)

FIG. 3. The main steps of the SOA.

and s is the population size. Then, the jth element of the ith seeker's position is updated by

$$x_{ij}(t+1) = x_{ij}(t) + d_{ij}\alpha_{ij}. \tag{4}$$

Since the subpopulations are searching using their own information, they are easy to converge to a local optimum. To avoid this situation, an inter-subpopulation learning strategy (illustrated by Fig. 4) is used; that is, the worst two positions of each subpopulation are combined with the best position of each of the other two subpopulations by the following binomial crossover operator:

$$x_{k_n j,\text{worst}} = \begin{cases} x_{lj,\text{best}} & \text{if} \quad R_j \leqslant 0.5, \\ x_{k_n j,\text{worst}} & \text{else}, \end{cases} \tag{5}$$

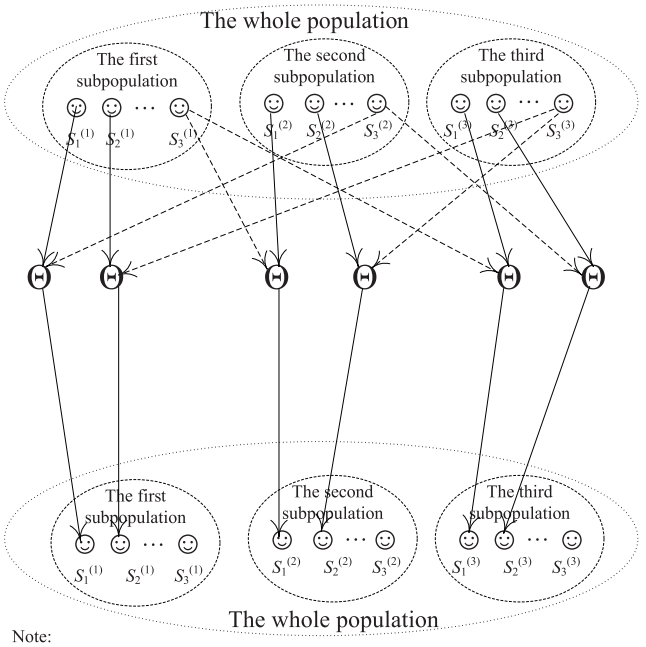
where $j = 1,2,\ldots,D$, R_j is a uniformly random real number within $[0,1]$, $x_{k_n j,\text{worst}}$ is denoted as the jth element of the nth worst position in the kth subpopulation, $x_{lj,\text{best}}$ is the jth element of the best position in the lth subpopulation, and the indices k, n, l are constrained by the combination $(k,n,l) \in \{(1,1,2),(1,2,3),(2,1,1),(2,2,3),(3,1,1),(3,2,2)\}$. In this way, the good information obtained by each subpopulation is exchanged among the subpopulations and then the diversity of the population is increased.

B. Search direction

The gradient has played an important role in the history of search methods [48]. The search space may be viewed as a gradient field [49], and a so-called empirical gradient (EG) can be determined by evaluating the response to the position change, especially when the objective function is not be available in a differentiable form at all [50]. The gradient can be defined as follows:

$$\vec{\nabla} J(\vec{x}) = \left(\frac{\partial J}{\partial x_1}, \frac{\partial J}{\partial x_2}, \ldots, \frac{\partial J}{\partial x_D} \right)^{\mathrm{T}}, \tag{6}$$

where $J(\cdot)$ denotes the objective function in the D-dimensional search space. Since most practical applications involve objective functions which cannot be available in a differentiable



Note:
1. $S_i^{(k)}$ denotes the ith seeker of the kth subpopulation;
2. Let $S_1^{(k)}$ and $S_2^{(k)}$ is the worst two seekers in the kth subpopulation (i.e., the objective function values of $S_1^{(k)}$ and $S_2^{(k)}$ are not smaller than that of any other seekers in the same subpopulation. Assume that the optimization problem to be solved is minimization problem), and $S_3^{(k)}$ is the best seeker in the kth subpopulation (i.e., the objective function value of $S_3^{(k)}$ is not larger than that of any other seekers in the same subpopulation) (k=1,2,3);
3. "Θ" denotes the binomial crossover operator
4. The inter-subpopulation learning rules:
$S_1^{(1)} \leftarrow S_1^{(1)} \Theta\, S_3^{(2)}$, $S_2^{(1)} \leftarrow S_2^{(1)} \Theta\, S_3^{(3)}$, $S_1^{(2)} \leftarrow S_1^{(2)} \Theta\, S_3^{(1)}$, $S_2^{(2)} \leftarrow S_2^{(2)} \Theta\, S_3^{(3)}$,
$S_1^{(3)} \leftarrow S_1^{(3)} \Theta\, S_3^{(1)}$, $S_2^{(3)} \leftarrow S_2^{(3)} \Theta\, S_3^{(2)}$

FIG. 4. The inter-subpopulation learning strategy.

form or their derivatives cannot be easily computed [51,52], a so-called EG [50] can be determined by evaluating the response to the position change. Then the seeker can follow an EG to guide his search. An EG is described as follows:

$$\vec{\nabla} J(\vec{x}) = \left(\frac{J(\vec{x}') - J(\vec{x})}{x_1' - x_1}, \frac{J(\vec{x}') - J(\vec{x})}{x_2' - x_2}, \ldots, \frac{J(\vec{x}') - J(\vec{x})}{x_D' - x_D} \right)^{\mathrm{T}}, \tag{7}$$

where $\vec{x}'$ and $\vec{x}$ are two different positions in search space.

Since the SOA does not involve the magnitude of the EG, search direction can be determined only by the signum function of a better position minus a worse position. For example, an empirical search direction,

$$\vec{d} = \begin{cases} \mathrm{sgn}(\vec{x}' - \vec{x}) & \text{if} \quad J(\vec{x}') < J(\vec{x}), \\ \mathrm{sgn}(\vec{x} - \vec{x}') & \text{else}, \end{cases} \tag{8}$$

where the function $\mathrm{sgn}(\cdot)$ is a signum function on each dimension of the input vector. In SOA, every seeker selects his search direction based on several EGs by evaluating the current or historical positions of himself or his neighbors. They are detailed as follows.

Swarms (including human beings) are a class of entities found in nature which specialize in mutual cooperation among them in executing their routine needs and roles [53]. There are two extreme types of co-operative behaviors. One, *egotistic*, is entirely proself and another, *altruistic*, is entirely progroup

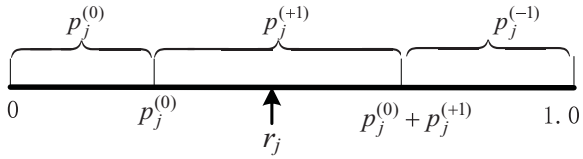


FIG. 5. The proportional selection rule of search directions [45].

[54]. Each seeker i, as a single sophisticated agent, is uniformly egotistic, believing that he should go toward his historical best position $\vec{p}_{i,\text{best}}(t)$ through cognitive learning [55]. Then, an EG from $\vec{x}_i(t)$ to $\vec{p}_{i,\text{best}}(t)$ can be involved for the ith seeker at time step t. Hence, each seeker i is associated with an empirical direction called an egotistic direction $\vec{d}_{i,\text{ego}}(t) = [d_{i1,\text{ego}}, d_{i2,\text{ego}}, \ldots, d_{iD,\text{ego}}]$:

$$\vec{d}_{i,\text{ego}}(t) = \text{sgn}[\vec{p}_{i,\text{best}}(t) - \vec{x}_i(t)]. \tag{9}$$

On the other hand, based on the altruistic behavior, the seekers in a same neighborhood region co-operate explicitly, communicate with each other, and adjust their behaviors in response to others to achieve the desired goal [54]. That means the seekers exhibit entirely progroup behavior through social learning [55]. The population then exhibits a self-organized aggregation behavior of which the positive feedback usually takes the form of attraction toward a given signal source [56]. For a "black-box" problem in which the ideal global minimum value is unknown, the neighbors' historical best position $\vec{g}_{\text{best}}(t)$ or the neighbors' current best position $\vec{l}_{\text{best}}(t)$ is used as the attraction signal source of the self-organized aggregation behavior. Hence, each seeker i is associated with two optional altruistic direction, that is, $\vec{d}_{i,\text{alt}_1}(t)$ and $\vec{d}_{i,\text{alt}_2}(t)$:

$$\vec{d}_{i,\text{alt}_1}(t) = \text{sgn}[\vec{g}_{\text{best}}(t) - \vec{x}_i(t)], \tag{10}$$

$$\vec{d}_{i,\text{alt}_2}(t) = \text{sgn}[\vec{l}_{\text{best}}(t) - \vec{x}_i(t)]. \tag{11}$$

Moreover, seekers enjoy the properties of proactiveness: Seekers do not simply act in response to their environment; they are able to exhibit goal-directed behavior [57]. In addition, future behavior can be predicted and guided by past behavior [58]. As a result, the seeker may be proactive to change his search direction and exhibit goal-directed behavior according to his past behavior. Hence, each seeker i is associated with an empirical direction called a proactiveness direction $\vec{d}_{i,\text{pro}}(t)$:

$$\vec{d}_{i,\text{pro}}(t) = \text{sgn}[\vec{x}_i(t_1) - \vec{x}_i(t_2)], \tag{12}$$

where $\vec{x}_i(t_1)$ and $\vec{x}_i(t_2)$ are the best and the worst one of the set $\{\vec{x}_i(t), \vec{x}_i(t-1), \vec{x}_i(t-2)\}$.

According to human rational judgment, the actual search direction of the ith seeker, $\vec{d}_i(t) = [d_{i1}, d_{i2}, \ldots, d_{iD}]$, is based on a compromise among the aforementioned four empirical

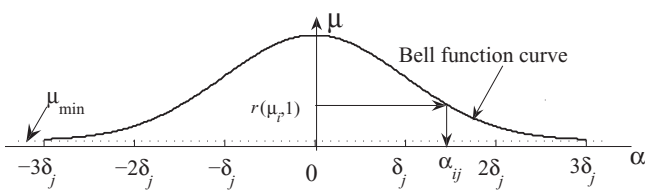


FIG. 6. The action part of the fuzzy reasoning [45].

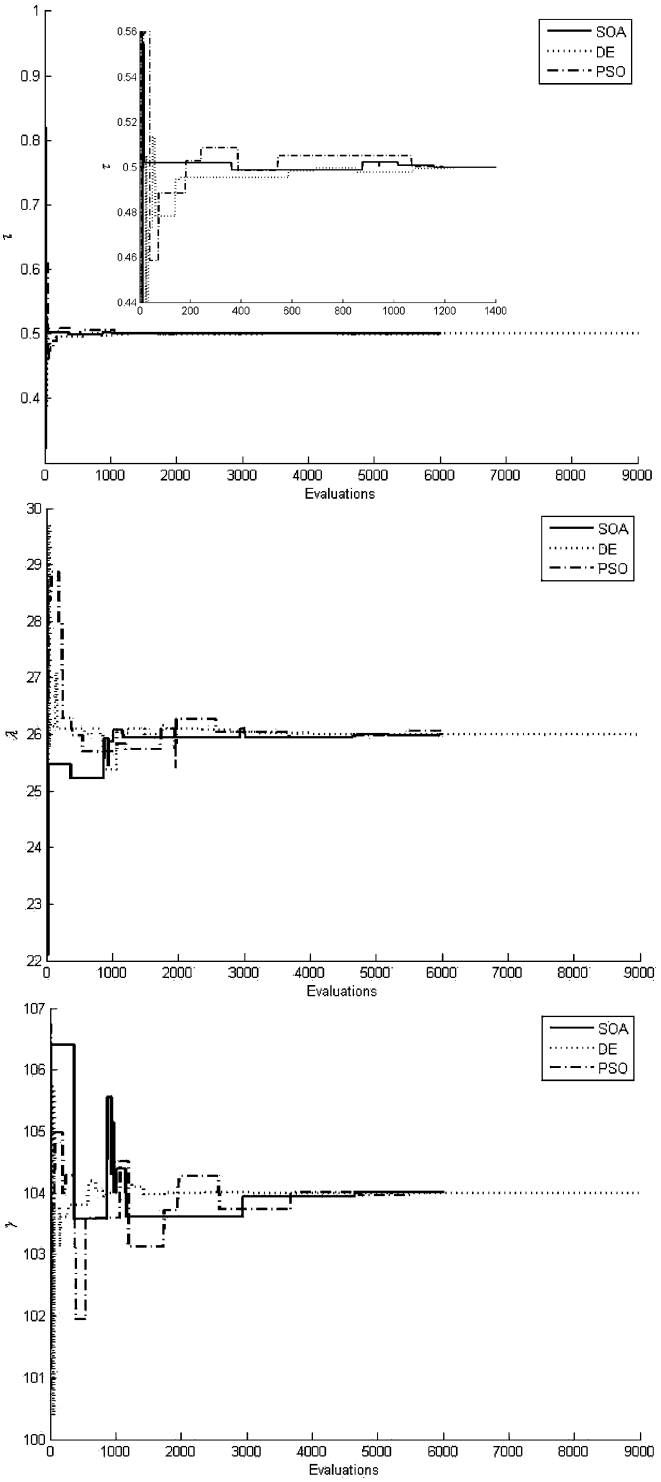


FIG. 7. Convergence graphs of the estimated parameters by various algorithms on time-delay logistic chaotic system.

directions, that is, $\vec{d}_{i,\text{ego}}(t)$, $\vec{d}_{i,\text{alt}_1}(t)$, $\vec{d}_{i,\text{alt}_2}(t)$, and $\vec{d}_{i,\text{pro}}(t)$. In this study, the jth element of $\vec{d}_i(t)$ is selected by applying the following proportional selection rule (shown in Fig. 5):

$$d_{ij} = \begin{cases} 0 & \text{if} \quad r_j \leqslant p_j^{(0)}, \\ 1 & \text{if} \quad p_j^{(0)} \leqslant r_j \leqslant p_j^{(0)} + p_j^{(1)}, \\ -1 & \text{if} \quad p_j^{(0)} + p_j^{(1)} \leqslant r_j \leqslant 1, \end{cases} \tag{13}$$

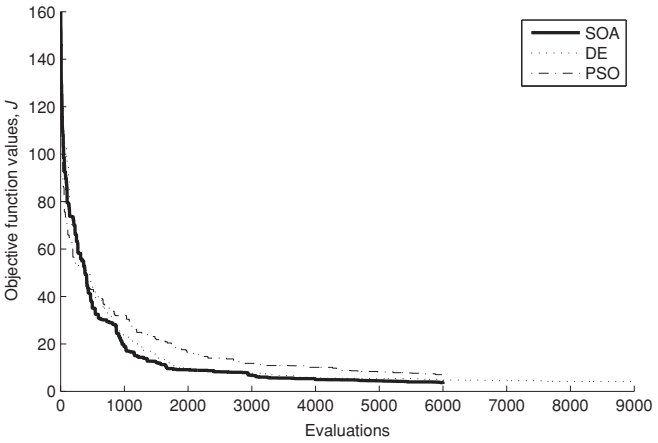


FIG. 8. Convergence graphs of various algorithms on time-delay logistic chaotic system (objective function values vs function evaluations).

where $i = 1,2,\ldots,s$, $j = 1,2,\ldots,D$, r_j is a uniform random number in [0, [1], and $p_j^{(m)}(m \in \{0,1,-1\})$ is defined as follows: In the set $\{d_{ij,\text{ego}}, d_{ij,\text{alt}_1}, d_{ij,\text{alt}_2}, d_{ij,\text{pro}}\}$, which is composed of the jth elements of $\vec{d}_{i,\text{ego}}(t), \vec{d}_{i,\text{alt}_1}(t), \vec{d}_{i,\text{alt}_2}(t)$, and $\vec{d}_{i,\text{pro}}(t)$, let $N^{(1)}$ be the number of "1", $N^{(-1)}$ be the number of "-1", and $N^{(0)}$ be the number of "0", then $p_j^{(m)} = \frac{N^{(m)}}{4}, m = 0,1,-1$. For example, if $d_{ij,\text{ego}} = 1$, $d_{ij,\text{alt}_1} = -1$, $d_{ij,\text{alt}_2} = -1$, $d_{ij,\text{pro}} = 0$, then $N^{(1)} = 1$, $N^{(-1)} = 2$, and $N^{(0)} = 1$. So, $p_j^{(1)} = \frac{1}{4}$, $p_j^{(-1)} = \frac{2}{4}$, and $p_j^{(0)} = \frac{1}{4}$.

C. Step length

In a continuous space, there often exists a neighborhood region close to an extremum point. In this region, the fitness values of the input variables are proportional to their distances from the extremum point. It may be assumed that better points are likely to be found in the neighborhood of families of good points, and search should be intensified in regions containing good solutions through focusing search [51]. Hence, from the standpoint on human searching, it could be believed that one may find the near optimal solutions in a narrower neighborhood of the point with lower fitness value and, contrariwise, in a wider neighborhood of the point with higher fitness value.

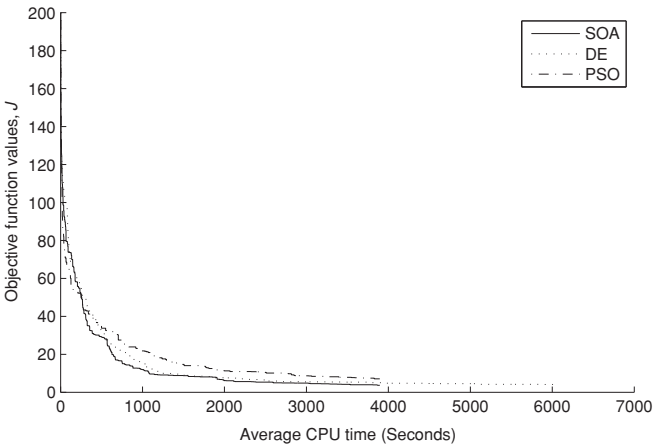


FIG. 9. Convergence graphs of various algorithms on time-delay logistic chaotic system (objective function values vs time).

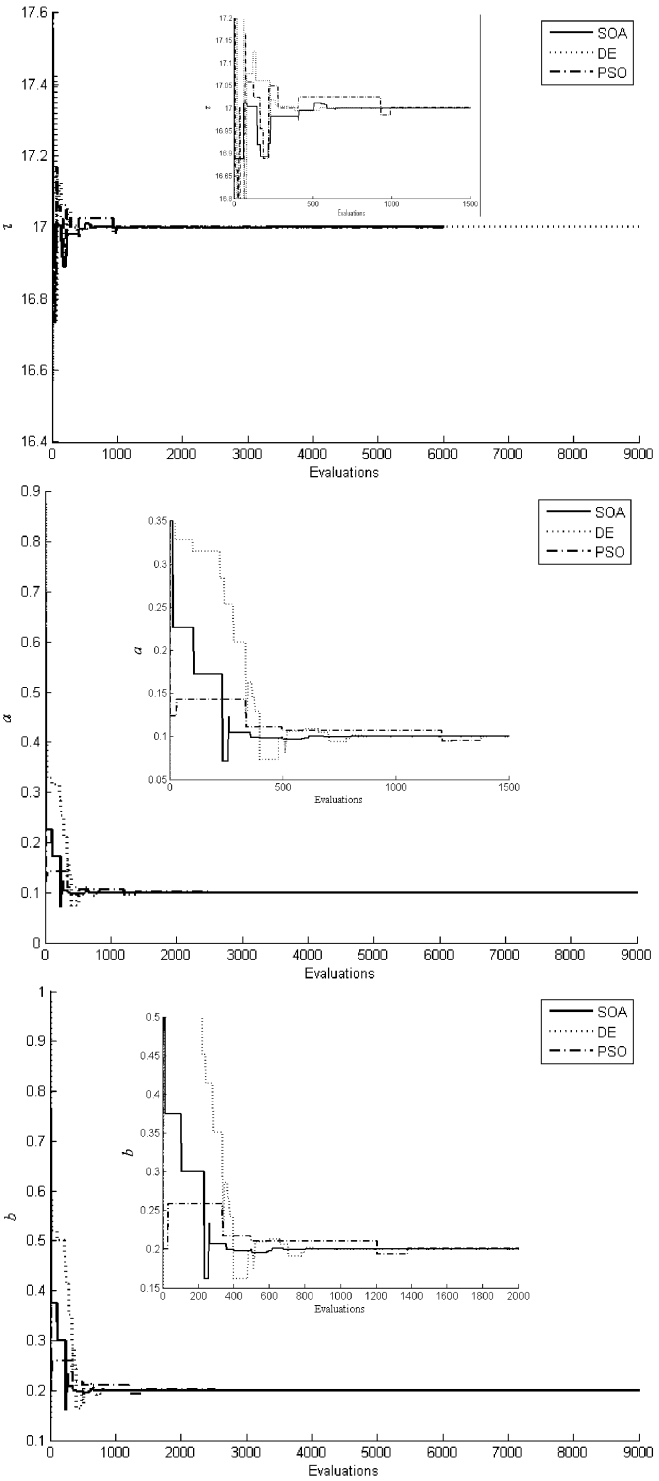


FIG. 10. Convergence graphs of the estimated parameters by various algorithms on time-delay Mackey-Glass chaotic system.

Fuzzy systems arose from the desire to describe complex systems with linguistic descriptions [59]. According to human focusing searching discussed above, the uncertainty reasoning of human searching could be described by natural linguistic variables and a simple control rule as "If *fitness value* is *small* (i.e., the conditional part), then *step length* is *short* (i.e., the

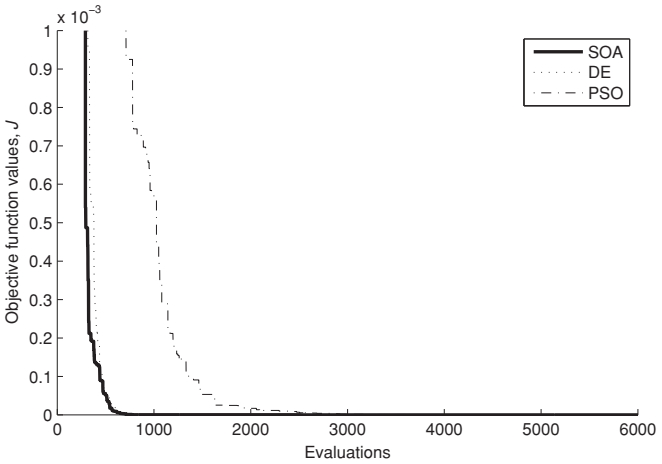


FIG. 11. Convergence graphs of various algorithms on time-delay Mackey-Glass chaotic system (objective function values vs function evaluations).

action part)." The understanding and linguistic description of human searching makes fuzzy system a good candidate for simulating human focusing searching behavior.

Different optimization problems often have different ranges of fitness values. To design a fuzzy system to be applicable to a wide range of optimization problems, the fitness values of all the seekers are descendingly sorted and turned into the sequence numbers from 1 to s as the inputs of fuzzy reasoning. The linear membership function is used in the conditional part (fuzzification) since the universe of discourse is a given set of numbers, that is, $1,2,\ldots,s$. The expression is presented as Eq. (14):

$$\mu_i = \mu_{\max} - (\mu_{\max} - \mu_{\min})\frac{s - I_i}{s - 1}, \tag{14}$$

where I_i is the sequence number of $\vec{x}_i(t)$ after sorting the fitness values and $\mu_{\max}$ is the maximum membership degree value which is equal to or a little less than 1.0. In this study, $\mu_{\max} = 0.99$.

In the action part (defuzzification), the Bell membership function $\mu(\alpha_{ij}) = e^{-\alpha_{ij}^2/(2\delta_j^2)}$ $(i = 1,2,\ldots,s;\ j = 1,2,\ldots,D)$ is used for the jth element of the ith seeker's step length. For the Bell function, the membership degree values of the input variables beyond $[-3\delta_j, 3\delta_j]$ are less than 0.0111 $[\mu(\pm 3\delta_j) = 0.0111]$, which can be neglected for a linguistic atom [60]. Thus, the minimum value $\mu_{\min} = 0.0111$ is set. Moreover, the parameter δ_j of the Bell membership function is the jth element of the vector $\vec{\delta} = [\delta_1, \delta_2, \ldots, \delta_D]$ which is given by

$$\vec{\delta} = \omega \mathrm{abs}(\vec{x}_{\mathrm{best}} - \vec{x}_{\mathrm{rand}}), \tag{15}$$

where the function abs(·) returns an output vector such that each element of the vector is the absolute value of the corresponding element of the input vector. The ω is a time-dependent weight used to decrease the step length with time step increasing so as to gradually improve the search precision. In this study, ω is linearly decreased from $\omega_{\max} = 0.9$ to $\omega_{\min} = 0.1$ during a run. The $\vec{x}_{\mathrm{best}}$ and $\vec{x}_{\mathrm{rand}}$ are the best seeker and a randomly selected seeker in the same subpopulation to which the ith seeker belongs, respectively. Notice that $\vec{x}_{\mathrm{rand}}$ is different from $\vec{x}_{\mathrm{best}}$, and $\vec{\delta}$ is shared by all the seekers in the same subpopulation. Then the *action* part of the fuzzy reasoning (shown in Fig. 6) gives the jth element of the ith seeker's step length $\vec{\alpha}_i = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{iD}](i = 1,2,\ldots,s;\ j = 1,2,\ldots,D)$:

$$\alpha_{ij} = \delta_j \sqrt{-\log[r(\mu_i, 1)]}, \tag{16}$$

where δ_j is the jth element of the vector $\vec{\delta}$ in Eq. (15), the function log(·) returns the natural logarithm of its input, and the function $r(\mu_i, 1)$ returns a uniform random number within the range of $[\mu_i, 1]$ which is used to introduce the randomness for each element of $\vec{\alpha}_i$ and improve local search capability.

TABLE I. The results of objective values for various algorithms on time-delay logistic chaotic system over 30 runs.

Algorithms	Best	Worst	Mean	Std.	h	CI
PSO	2.3273	6.1792	4.1725	2.4259	0	[−2.3449, 0.2547]
DE	2.0647	6.8942	4.0519	1.5117	0	[−2.2188, 0.3698]
SOA	1.5584	5.1601	3.1274	1.2675		

IV. SIMULATION RESULTS

To evaluate the effectiveness and efficiency of the proposed SOA-based parameter-estimation approach for time-delay chaotic system, two typical chaotic systems are used as the test systems which were employed by Refs. [27,28]. Moreover, the proposed SOA is compared with two state-of-the-art algorithms, that is, PSO [27] and DE [28]. These two algorithms were used to solve the same parameter-estimation problems of a time-delay chaotic system by Refs. [27,28]. All the algorithms are implemented in MATLAB 7.0 and run on a PC with Pentium 4 CPU 2.4G 512MB RAM. The implementations of both PSO and DE follow the suggestions from Refs. [27,28], respectively. The parameters of PSO are set as follows: The population size is equal to 20, learning rate $c_1 = c_2 = 2$, inertia weight linearly decreased from 0.9 to 0.4 with run time increasing, the maximum velocity $v_{\max}$ is set at 20% of the dynamic range of the variable on each dimension. The control parameters of DE algorithm are set as follows: DE/rand-best/1/bin, the population size is 30, mutation factor F and crossover rate CR are set to 0.6. The main parameters involved in SOA include the population size s

TABLE II. The estimated parameters for various algorithms on time-delay logistic chaotic system over 30 runs.

Algorithms		τ	γ	λ
PSO	Best	0.4997	25.9999	104.0000
	Worst	0.5004	25.8301	103.6808
	Mean	0.4998	25.9654	103.9469
DE	Best	0.5003	26.0607	104.0114
	Worst	0.4978	26.0000	104.0000
	Mean	0.4996	25.9999	104.0001
SOA	Best	0.5000	25.9962	104.0136
	Worst	0.4999	26.1262	103.9873
	Mean	0.4997	26.0286	104.0014

TABLE III. The CPU time necessary to simulate 300 generations of various algorithms on time-delay logistic chaotic system over 30 runs (for example, the shortest time needed by PSO is 3.2907×10^3 s).

Algorithms	Shortest time	Longest time	Average time
PSO	3.2907×10^3	3.3653×10^3	3.3239×10^3
DE	5.0626×10^3	5.3011×10^3	5.1475×10^3
SOA	3.1786×10^3	3.3834×10^3	3.3153×10^3

and the parameters of membership function of fuzzy reasoning [including the limits of membership degree value, that is, $\mu_{\max}$ and $\mu_{\min}$ in Eq. (14) and the limits of ω, that is, $\omega_{\max}$ and $\omega_{\min}$ in Eq. (15)]. In this paper, $s = 20$, $\mu_{\max} = 0.99$, $\mu_{\min} = 0.0111$, $\omega_{\max} = 0.9$, $\omega_{\min} = 0.1$ for both the test systems. Total 30 runs and the maximum generations of 300 are made for all the algorithms. The following conclusions drawn from the comparisons are valid at least for the corresponding versions of the three algorithms with their above-selected parameters. In the simulations, the original system first evolves freely with a random initial state. After a period of transient process, a state is selected as the initial state $\mathbf{X}_0$ for parameter estimation. Five hundred states are used to calculate the objective function in Eq. (3). Like Ref. [28], dde23 in MATLAB Toolbox is used to obtain the state vectors $\boldsymbol{X}$ and $\tilde{\boldsymbol{X}}$ at each time t.

A. Simulation on time-delay logistic chaotic system

The first test example is time-delay logistic chaotic system described as

$$\dot{\boldsymbol{Y}}(t) = -\lambda \boldsymbol{Y}(t) + \gamma \boldsymbol{Y}(t-\tau)[1 - \boldsymbol{Y}(t-\tau)], \qquad (17)$$

where τ, γ, and λ are unknown parameters to be estimated. The original system is assigned with the original parameters: $\tau = 0.5$, $\gamma = 104$, and $\lambda = 26$, under which the system is chaotic. The search ranges of three parameters are set as $0.2 \leqslant \tau \leqslant 1$, $100 \leqslant \gamma \leqslant 108$, and $22 \leqslant \lambda \leqslant 30$.

To compare the proposed method with other algorithms, the concerned performance indexes including the "Best," "Worst," "Mean," and standard deviation ("Std.") of the objective function values are summarized in Table I. In order to determine whether the results obtained by SOA are statistically different from the results generated by other algorithms, the two-sample t test [61,62] are conducted which performs a t test of the null hypothesis that two independent random samples have equal means and equal but unknown variances, against the alternative that the means are not equal. An h value of one indicates that the performances of the two algorithms are statistically different with 95% certainty, whereas h value of zero implies that the performances are not statistically different. The CI is confidence interval on the difference of sampled data means. (Here are the means of the final minimum objective function values obtained by the two considered algorithms over 30 runs.) The corresponding h and CI values are presented in Table I, too. According to the respective Best or Worst objective function values, the Best, Worst, and Mean values of the estimated parameters by various algorithms are listed in Table II. The total CPU time necessary to simulate 300 generations of each algorithm in a run is summarized in Table III. The convergence graphs of the estimated parameters by various algorithms are illustrated in Fig. 7. The average convergence curves for objective function values vs number of function evaluations and objective function values vs CPU time are depicted for all the algorithms in Figs. 8 and 9, respectively. Note that function evaluations correspond to each time the objective function in Eq. (3) is evaluated in the iteration process of the algorithm, and the maximum generations multiplied by the population size is the maximum number of function evaluations (please refer to Fig. 3 to understand it).

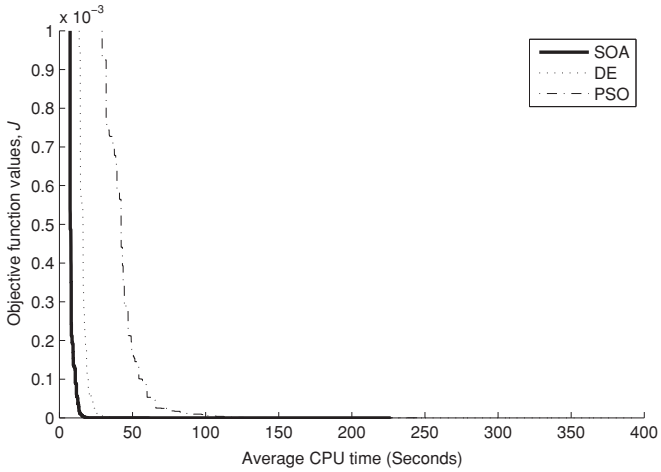


FIG. 12. Convergence graphs of various algorithms on time-delay Mackey-Glass chaotic system (objective function values vs time).

Table I indicates that SOA has the smallest Best, Mean, Worst, and Std. values of objective function of all the other listed algorithms. Although the h value for both SOA vs PSO and SOA vs DE is equal to zero, the median values of the corresponding CI are less than zero. Hence, the conclusions can be drawn that SOA has, at least, equally good performance as both DE and PSO do according to the h values. In addition, according to its smallest Best, Mean, Worst, and Std. values, as well as the median values of the corresponding CI less than zeros, the SOA is better than the other two listed algorithms in terms of global search capacity and local search precision. From Table III, it can be seen that the computing time

TABLE IV. The results of objective values for various algorithms on time-delay Mackey-Glass chaotic system over 30 runs (For example, the best objective function value over 30 runs of PSO is 1.3801×10^{-13}).

Algorithms	Best	Worst	Mean	Std.	h	CI
PSO	1.38×10^{-13}	6.68×10^{-9}	1.30×10^{-9}	2.20×10^{-9}	0	$[-2.77 \times 10^{-9}, 1.58 \times 10^{-10}]$
DE	0	0	0	0		
SOA	0	0	0	0		

TABLE V. The estimated parameters for various algorithms on time-delay Mackey-Glass chaotic system over 30 runs.

Algorithms		τ	a	b
PSO	Best	$17 - 3.85 \times 10^{-13}$	$0.1 - 4.19 \times 10^{-14}$	$0.2 - 1.50 \times 10^{-14}$
	Worst	$17 + 1.82 \times 10^{-11}$	$0.1 + 3.62 \times 10^{-11}$	$0.2 + 9.39 \times 10^{-12}$
	Mean	$17 + 3.79 \times 10^{-12}$	$0.1 + 1.92 \times 10^{-12}$	$0.2 + 3.45 \times 10^{-12}$
DE	Best	17	0.1	0.2
	Worst	17	0.1	0.2
	Mean	17	0.1	0.2
SOA	Best	17	0.1	0.2
	Worst	17	0.1	0.2
	Mean	17	0.1	0.2

of SOA is less than that of other algorithms. Furthermore, Figs. 8 and 9 show that SOA has faster convergence speed and needs not only less CPU time but also fewer function evaluations to achieve the corresponding precision levels of both PSO and DE. In Fig. 7, the estimated parameters (i.e., τ, γ, and λ) by the three algorithms at every function evaluation are depicted. Generally, if the convergence graphs of the estimated parameters have a serious vibration but a large objective function value, it means that the considered algorithm has a good exploration ability but a bad exploitation ability; contrarily, if the convergence graphs of the estimated parameters have a little or no vibration but a large objective function value, it means that the considered algorithm may get trapped in a local optimum (i.e., premature convergence) or search stagnation. On the one hand, the SOA has a smaller objective function value than both PSO and DE according to Table I. On the other hand, according to Fig. 7, the convergence graphs of the estimated parameters by the SOA has a more serious vibration than that of DE and a less vibration than that of PSO. Hence, it can be seen that the SOA has a better trade-off between exploration and exploitation.

B. Simulation on time-delay Mackey-Glass chaotic system

To further evaluate the proposed method, the time-delay Mackey-Glass chaotic system is used, which is described as

$$\dot{\boldsymbol{Y}}(t) = -a\boldsymbol{Y}(t) + \frac{b\boldsymbol{Y}(t-\tau)}{1+[\boldsymbol{Y}(t-\tau)]^{10}}, \tag{18}$$

where τ,a and b are unknown parameters to be estimated. The original system is assigned with the original parameters: $\tau = 17$, $a = 0.1$, and $b = 0.2$ under which the system is chaotic. The search ranges of three parameters are set as $12 \leqslant \tau \leqslant 20$, $0.05 \leqslant a \leqslant 1$, and $0.05 \leqslant b \leqslant 1$.

The corresponding simulation results over all 30 runs are summarized in Tables IV–V. The convergence graphs of the estimated parameters by various algorithms are illustrated in Fig. 10. The average convergence curves for objective function values vs number of function evaluations and objective function values vs CPU time are depicted for all the algorithms in Figs. 11 and 12, respectively.

From Tables IV and V, it can be seen that both SOA and DE can successfully find the true parameters and achieve the precise objective function value $J = 0$ in all the 30 runs. Because of their same results, it is not necessary to perform the t tests on SOA vs DE. Although the h value for SOA vs PSO equals zero, the median value of the corresponding CI,-2.61×10^{-9}, is less than zero. In particular, PSO always cannot exactly find the true parameters in every run with an objective function value larger than zero. In Table VI and Figs. 11 and 12, it is shown that SOA also has faster convergence speed and needs less time to achieve the objective function value of zero than the other methods. Moreover, Fig. 10 also shows that the parameters estimated by SOA converge to a steady state of the optimal solutions (i.e., the original parameters of the original test system) with a faster speed than that of the other algorithms. In the simulations, PSO cannot achieve the objective function value of zero within the maximum generations of 300 at every run, DE on average needs 5642 function evaluations and consumes CPU time of

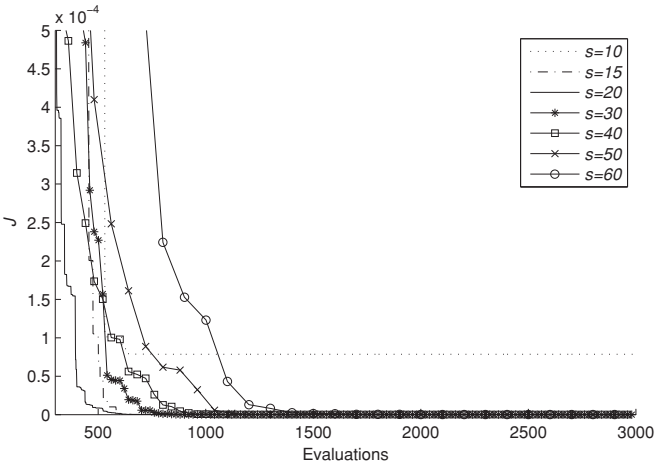


FIG. 13. Convergence graphs with various population sizes.

TABLE VI. Computing time (s) for various algorithms on time-delay Mackey-Glass chaotic system over 30 runs.

Algorithms	Shortest time	Longest time	Average time
PSO	203.71	230.17	220.18
DE	343.33	371.05	354.40
SOA	193.74	222.29	203.98

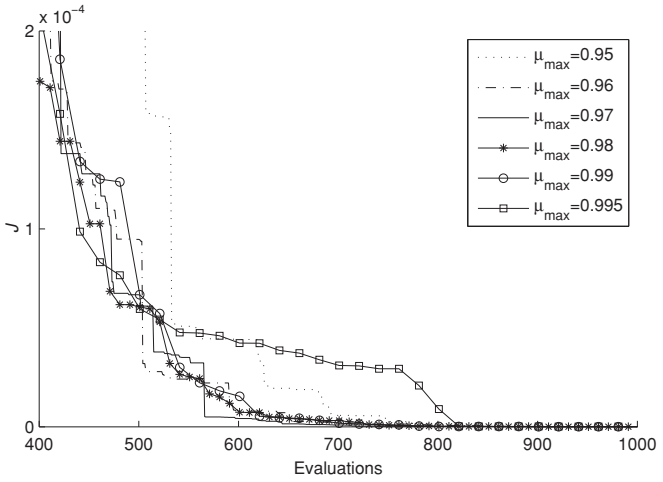


FIG. 14. Convergence graphs with various $\mu_{\max}$.

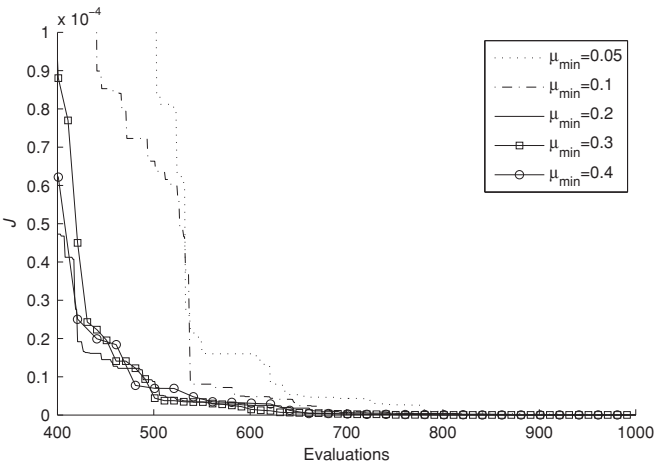


FIG. 15. Convergence graphs with various $\mu_{\min}$.

2.5634×10^2 s to obtain a mean objective function value of zero, but SOA needs only 4078 evaluations and 1.0015×10^2 s. Hence, the used version of PSO is obviously inferior to both DE and SOA because it cannot find the precise optimal solutions. Moreover, compared with DE, SOA needs less computation cost to find the precise optimal solutions with a better global convergence performance.

V. ANALYSIS ON THE PARAMETERS OF SOA

The main parameters in SOA include the population size s, the number of subpopulations K, and the parameters of membership function of fuzzy reasoning [including the limits of membership degree value, that is, $\mu_{\max}$ and $\mu_{\min}$ in Eq. (14) and time-dependent weight ω in Eq. (15)]. In this section, on the one hand, we presented the qualitative analysis on the effects of these parameters on the algorithm's performance. On the other hand, the quantitative study was provided on the time-delay Mackey-Glass chaotic system.

A. Population size

Generally, small population size shall partly decrease the diversity of the population, and big population size shall, on the contrary, increase the diversity of the population. However, a population size that is too big may deteriorate the algorithm performance because, in this case, many seekers may have not enough chance to learn from the global historical best seeker, which may result in a weak global convergence ability. Table VII presented the simulation results of SOA with different population size s on the time-delay Mackey-Glass chaotic system. From Table VII, it can be seen that a population size that is too small will deteriorate the performance of SOA. When population size is smaller than 15, the SOA is apt to get trapped in local minima. On the contrary, Fig. 13 also shows that the convergence speed may slow down when increasing population size to a certain value that is too big.

TABLE VII. The results of objective values for SOA with various population size over 30 runs.

s	10	15	20	30	40	50	60
Best	1.2326×10^{-32}	0	0	0	0	0	0
Worst	6.0184×10^{-4}	3.1131×10^{-20}	0	0	0	0	0
Mean	7.8570×10^{-5}	3.8913×10^{-21}	0	0	0	0	0
Std.	1.9271×10^{-4}	1.1006×10^{-20}	0	0	0	0	0

B. The limits of membership degree

From Eqs. (14) and (16), search step decreases with increasing $\mu_{\max}$ and $\mu_{\min}$, and step length equals to zero when $\mu_{\max} = 1.0$. Decreasing search step may result in good search precision, and increasing search step may speed up the convergence rate. However, a search step that is too small may result in slow convergence rate, and even premature convergence. On the contrary, a search step that is too big may result in the dissatisfactory convergence precision and even stagnation problem. Hence, both $\mu_{\max}$ and $\mu_{\min}$ may have a certain impact on the performance of SOA. Since the membership degree values of the input variables beyond $[-3\delta, 3\delta]$ are less than 0.0111 $[\mu(\pm 3\delta) = 0.0111]$, and the minimum value $\mu_{\min} = 0.0111$ is assigned. In order to keep a certain search ability of the best seeker, the maximum membership degree value is set to a little less than 1.0. Figures 14 and 15 present the simulation results of SOA with different $\mu_{\max}$ and $\mu_{\min}$, respectively. From the figures, it can be seen that the performance of SOA is not drastically sensitive to the change of the $\mu_{\max}$ and $\mu_{\min}$ values. Moreover, SOA can find the optimal solution in each run with the objective function value of zero. Of course, a suitable set of $\mu_{\max}$ and $\mu_{\min}$ still can obtain a faster convergence speed.

TABLE VIII. The results of objective values for SOA with various subpopulation number over 30 runs.

K	1	2	3	4	5
Best	0	0	0	0	3.6181×10^{-23}
Worst	1.0184×10^{-31}	0	0	6.1629×10^{-32}	2.0895×10^{-6}
Mean	6.1630×10^{-32}	0	0	5.6027×10^{-33}	5.2998×10^{-7}
Std.	2.9242×10^{-32}	0	0	1.8582×10^{-32}	8.3284×10^{-7}

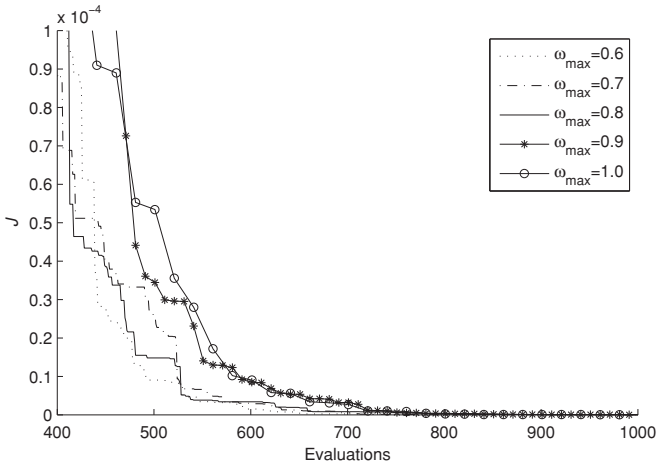


FIG. 16. Convergence graphs with various $\omega_{\max}$.

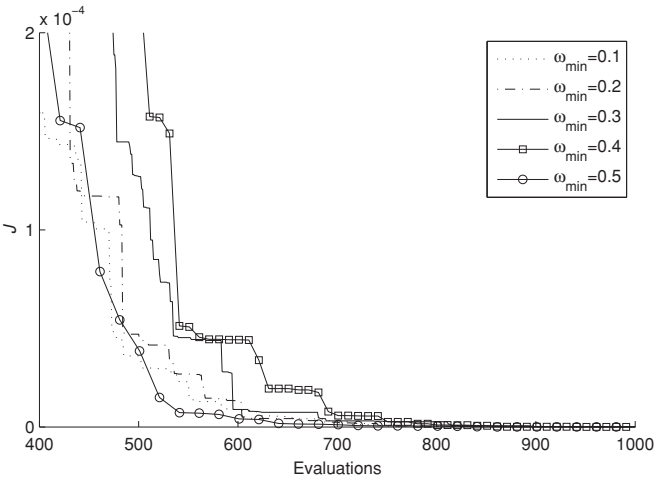


FIG. 17. Convergence graphs with various $\omega_{\min}$.

C. The limits of time-dependent weight

The ω in Eq. (15) is a time-dependently declined parameter used to decrease the step length with time step increasing so as to gradually improve the search precision. Like the membership degree in Sec. V B, this parameter throws the same influence on the performance of the SOA by tuning the step length. Figures 16 and 17 present the simulation results of SOA with different $\omega_{\max}$ and $\omega_{\min}$, respectively. However, from the figures, it can be seen that SOA can find the optimal solution in each run and the performance of SOA is not drastically sensitive to the change of the ω limits.

D. The number of subpopulations

In Sec. III, the number of subpopulations K is set as $K = 3$. In fact, this is also a key parameter of the algorithm and its effect should be discussed. When the whole population size is fixed, the greater the number of subpopulations, the smaller is the subpopulation size. Under this condition, although the large subpopulation number may increase the diversity of the population, the small subpopulation size results in the case in which the subpopulation does not have an intensive search and the whole population does not have a good global convergence ability. Table VIII presents the simulation results of SOA with different K. From the table, it can be seen that the performance of the SOA will be deteriorated with the increase of the K value.

VI. CONCLUSIONS

SOA is a novel heuristic stochastic optimization algorithm based on simulating human behaviors. The algorithm has the additional advantage of being easy to understand and simple to implement so that it can be used for a wide variety of optimization tasks. In this paper, a SOA-based approach for parameter estimation of time-delay chaotic systems is proposed, and the validity of SOA is studied. The simulation results show that SOA has better performance in balancing global search ability and convergence speed than other algorithms. So it is believed that the proposed SOA approach is capable of efficiently and effectively estimating the parameters of time-delay chaotic systems and will become a promising candidate for parameter-estimation problems. The future work is to apply the approach to dynamical chaotic systems, uncertain chaotic systems, and noise-perturbed chaotic systems.

ACKNOWLEDGMENTS

The authors are very grateful to Dr. Rainer Hegger for his constructive help. The authors would like to thank the reviewers for their helpful comments, too. This work is in part supported by the Fundamental Research Funds for the Central Universities (Grants No. SWJTU09CX026, No. SWJTU09ZT13, and No. SWJTU09ZT10), the National Natural Science Foundation of China (Grant No. 60870004), the National Basic Research Program of China (973 Program) (Grant No. 2007CB310704), the Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20100005110002), and the Program for New Century Excellent Talents in University of the Ministry of Education of China (Grant No. NCET-10-0239).

[1] U. Parlitz, L. Junge, and L. Kocarev, Phys. Rev. E **54**, 6253 (1996).
[2] U. Parlitz, Phys. Rev. Lett. **76**, 1232 (1996).
[3] A. Maybhate and R. E. Amritkar, Phys. Rev. E **59**, 284 (1999).
[4] A. Maybhate and R. E. Amritkar, Phys. Rev. E **61**, 6461 (2000).
[5] S. H. Chen and J. H. Lu, Phys. Lett. A **299**, 353 (2002).
[6] A. Sitz, U. Schwarz, J. Kurths, and H. U. Voss, Phys. Rev. E **66**, 016210 (2002).
[7] X. G. Wu and H. P. Hu, Chaos, Solitions Fractals **22**, 359 (2004).
[8] V. F. Pisarenko and D. Sornette, Phys. Rev. E **69**, 036122 (2004).
[9] D. Huang, Phys. Rev. E **69**, 067201 (2004).

[10] U. S. Freitas, Elbert E. N. Macau, and C. Grebogi, Phys. Rev. E **71**, 047203 (2005).
[11] Z. M. Ge and J. W. Cheng, Chaos, Solitions Fractals **24**, 597 (2005).
[12] L. L. Huang, M. Wang, and R. P. Feng, Phys. Lett. A **342**, 299 (2005).
[13] P. I. Marion and J. Miguez, Phys. Lett. A **351**, 262 (2006).
[14] W. Yu, G. Chen, J. Cao, J. Lu, and U. Parlitz, Phys. Rev. E **75**, 067201 (2007).
[15] R. H. Li, W. Xu, and S. Xi, Phys. Lett. A **367**, 199 (2007).
[16] H. D. I. Abarbanel, D. R. Creveling, and J. M. Jeanne, Phys. Rev. E **77**, 016208 (2008).
[17] A. A. Zaher, Phys. Rev. E **77**, 036212 (2008).
[18] D. Yu and U. Parlitz, Phys. Rev. E **77**, 066221 (2008).
[19] D. Ghosh and S. Banerjee, Phys. Rev. E **78**, 056211 (2008).
[20] J. C. Quinn, P. H. Bryant, D. R. Creveling, S. R. Klein, and H. D. I. Abarbanel, Phys. Rev. E **80**, 016201 (2009).
[21] H. Peng *et al*., Chaos **19**, 033130 (2009).
[22] Q. He, L. Wang, and B. Liu, Chaos, Solitions & Fractals **34**, 654 (2007).
[23] W. D. Chang, Chaos, Solitions & Fractals **32**, 1469 (2007).
[24] L. Q. Shen and M. Wang, Chaos, Solitions & Fractals **38**, 106 (2008).
[25] B. Peng *et al*., Chaos, Solitions & Fractals **39**, 2110 (2009).
[26] L. Wang and L. P. Li, Expert Syst. Appl. **37**, 1279 (2010).
[27] Y. Tang and X. Guan, Chaos, Solitions & Fractals **40**, 1391 (2009).
[28] Y. Tang and X. Guan, Chaos, Solitions & Fractals **42**, 3132 (2009).
[29] M. C. Mackey and L. Glass, Science **197**, 287 (1977).
[30] M. J. Bunner and W. Just, Phys. Rev. E **58**, R4072 (1998).
[31] M. J. Bunner *et al*., Phys. Lett. A **211**, 345 (1996).
[32] M. J. Bunner, Th. Meyer, A. Kittel, and J. Parisi, Phys. Rev. E **56**, 5083 (1997).
[33] M. J. Bunner *et al*., Eur. Phys. J. D **10**, 165 (2000).
[34] C. D. Li, X. F. Liao, and R. Zhang, Chaos, Solitions Fractals **23**, 1177 (2005).
[35] Y. Chen, X. X. Chen, and S. S. Gu, Nonlinear Anal. **66**, 1929 (2007).
[36] W. B. Langdon and P. Riccardo, Trans. Evolut. Comput. **11**, 561 (2007).
[37] J. Brest *et al*., IEEE Trans. Evolut. Comput. **10**, 646 (2006).
[38] L. dos Santos Coelho and B. M. Herrera, IEEE Trans. Ind. Electron. **54**, 3234 (2007).
[39] S. H. Ling *et al*., IEEE Trans. Ind. Electron. **55**, 3447 (2008).
[40] K. Mathias, Ph.D. thesis, Department of Computer Science, University of Essex, 2006.
[41] P. A. Donald, *Optimization Theory with Applications* (Dover, New York, 1986).
[42] C. Dai *et al*., J. Syst. Eng. Electron. **21**, 300 (2010).
[43] C. Dai *et al*., Int. J. Electr. Power Energy Syst. **33**(3), 369 (2011).
[44] C. Dai *et al*., Electr. Power Syst. Res. **79**, 1462 (2009).
[45] C. Dai *et al*., IEEE Trans. Power Syst. **24**, 1218 (2009).
[46] C. Dai, W. R. Chen, and Y. F. Zhu, IEEE Trans. Ind. Electron. **57**, 1710 (2010).
[47] J. D. Hewlett, B. M. Wilamowski, and G. Dundar, IEEE Trans. Ind. Electron. **55**, 3374 (2008).
[48] N. Hansen, *The CMA Evolution Strategy: A Tutorial* [http://www.lri.fr/∼hansen/].
[49] L. Steels, in *Decentralized AI: Proceedings of the First European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW-89)*, edited by Y. Dernazean and J. P. Muller (Elsevier, Amsterdam, 1990), pp. 175–196.
[50] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. (Pearson Education Asia Ltd. and Tsinghua University Press, Beijing, 2006).
[51] B. Raphael and I. F. C. Smith, Appl. Math. Comput. **146**, 729 (2003).
[52] M. H. Wright, in *Numerical Analysis 1995*, edited by D. F. Griffiths and G. A. Watson (Addison-Wesley Longman, Reading, MA, 1996), pp. 191–208.
[53] V. Kumar and F. Sahin, Proc. IEEE Int. Conf. Syst. Man Cybernet. **4**, 3364 (2003).
[54] D. Eustace, D. P. Barnes, and J. O. Gray, Proc. Int. Conf. Ind. Electron. Control Instrum. **1**, 39 (1993).
[55] Y. del Valle *et al*., IEEE Trans. Evol. Comput. **12**, 171 (2008).
[56] V. Trianni *et al*., in *Evolving Aggregation Behaviors in a Swarm of Robots*, edited by W. Banzhaf *et al*., Lecture Notes in Artificial Intelligence (Springer-Verlag, Berlin, 2003), Vol. 2801, pp. 865–874.
[57] M. Wooldridge and N. R. Jennings, Knowl. Eng. Rev. **10**, 115 (1995).
[58] I. Ajzen, Pers. Soc. Psychol. Rev. **6**, 107 (2002).
[59] L. A. Zadeh, Inf. Science **8**, 199 (1975).
[60] D. Li, Comput. Math. Appl. **35**, 99 (1998).
[61] MATLAB *Statistics Toolbox User's Guide* [http://www.mathworks.com/help/pdf_doc/stats/stats.pdf].
[62] A. Agresti and C. Franklin, *Statistics: The Art and Science of Learning from Data*, 2nd ed. (Prentice Hall, Englewood Cliffs, NJ, 2007).