Journal of AI and Data Mining
Vol 8, No 3, 2020, 303-312.
DOI: 10.22044/JADM.2020.8594.1993

Chaotic-based Particle Swarm Optimization with Inertia Weight for Optimization Tasks

N. Mobaraki[1], R. Boostani[2*] and M. Sabeti[3]

1. Department of Computer Engineering, Apadana Institute of Higher Education, Shiraz, Iran.
2. Department of CSE & IT, Faculty of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran.
3. Department of Computer Engineering, North Tehran Branch, Islamic Azad University, Tehran, Iran.

Received 18 June 2019; Revised 22 February 2020; Accepted 15 April 2020
**Corresponding author: boostani@shirazu.ac.ir (R. Boostani).*

Abstract
Among a variety of meta-heuristic population-based search algorithms, particle swarm optimization (PSO) with adaptive inertia weight (AIW) has been considered as a versatile optimization tool, which incorporates the experience of the whole swarm into the movement of particles. Although the exploitation ability of this algorithm is great, it cannot comprehensively explore the search space and may be trapped in a local minimum through a limited number of iterations. To increase its diversity as well as enhance its exploration ability, this paper inserts a chaotic factor, generated by three chaotic systems, along with a perturbation stage into AIW-PSO to avoid premature convergence, especially in complex non-linear problems. To assess the proposed method, a known optimization benchmark containing non-linear complex functions is selected and its results are compared with those of standard PSO, AIW-PSO, and genetic algorithm (GA). The empirical results demonstrate the superiority of the proposed chaotic AIW-PSO to the counterparts over 21 functions, which confirms the promising role of inserting the randomness into AIW-PSO. The behavior of error through the epochs show that the proposed manner can smoothly find proper minima in a timely manner without encountering a premature convergence.

Keywords: *PSO-AIW, Randomness, Chaotic Factor, Swarm Experience, Convergence Rate.*

1. Introduction

In the last two decades, the rate of employing meta-heuristic search schemes in the optimization-based applications has drastically increased. Since most optimization problems do not have an exact analytical solution, the demand for employing heuristic search methods for optimizing model parameters has drastically grown. Moreover, these searching methods can be executed in the case of accessing just a limited number of samples, while the theoretical estimation methods require plenty of samples to support the validity of the achieved results.

The primary heuristic methods like Tabu search [1] and simulated annealing [2, 3] are the single agent search methods, while the newer heuristic methods termed as meta-heuristic methods try to find the extremum points by emitting a population of agents in order to augment the exploration property over the search space. As the amount of randomness in creating a new generation has increased (e.g. genetic algorithm (GA) [4, 5], the exploration ability of the algorithm has been improved, while the convergence rate has been significantly diminished. In order to compromise between the convergence rate and the searching ability, particle swarm optimization (PSO) [6-9] has been developed to create a population of particles, which are randomly allocated to the search space, and move each particle toward the sample with the highest fitness (gbest), while considering its best experience (pbest) in each epoch, thanks to PSO for its fast exploitation in the search space, compared to the known meta-heuristic methods like ant colony optimization (ACO) and GA. The fast convergence of PSO is the result of low randomness in the displacement of particles with a constant velocity and the acceleration parameters [10, 11]. Nonetheless, PSO suffers from a low diversity among the particles, leading to diminish its exploration capability, and is mostly

encountered premature convergence, especially in high-dimensional spaces. To increase the randomness of particles' movement, it is repeatedly tried to insert a chaotic factor to avoid a premature convergence [12, 13]. Chaotic systems are rule-based systems and their input-output mathematical relations are accessible. The output of chaotic systems behaves randomly, though they obey certain analytical-based relations. This is because describing the relation of these systems is extremely sensitive to their initial conditions, and also their input-output formula are mostly either a 3rd order (or higher than 3) non-linear differential equation or even can be a low-order time-varying differential equation. In general, rendering long-term prediction is impossible for chaotic systems, while their outputs are predictable within short intervals of time. The output of these systems is limited between two values (hence, they are stable), which makes them suitable for use as a random factor to deteriorate the movement path of particles to enhance their exploration ability.

In this regard, Hosseinpourfard *et al.* [14] have proposed a chaotic particle swarm optimization (CPSO) that employs a Lorenz system, Tent map, and Henon map to generate random numbers used in the update position formula of PSO. Their experimental results have shown an improvement in the optimization ability compared to the standard PSO at the cost of a bit lower convergence. In addition, CPSO has outperformed both standard GA and chaotic GA (CGA) [15]. Moreover, this randomness property has been repeatedly used for image encryption [16]. It is, therefore, these seemingly random sequence of numbers that can be exactly regenerated in the receiver station and the original image can be finely decoded. The chaotic-based encryption techniques have obtained a high level of robustness against brute-force and statistical invasions.

Wang *et al.* [17] have used GA and PSO in conjunction with a chaotic function to overcome the premature convergence of PSO as well as weak exploitation of GA. Their experimental results over five classic benchmark functions have shown that their proposed hybrid method significantly outperforms the standard GA and PSO in terms of global precision and convergence rate. Yang *et al.* [18] have considered an improved logistic map (double-bottom map) for PSO, called DBM-PSO, to compromise between the exploration and exploitation properties. Their experimental results over 22 benchmark functions indicate that the performance of DBM-PSO is significantly better than the performance of other PSOs. Li *et al.* [19] have proposed an effective chaos-based optimization algorithm (COA), yielding a much higher performance than that of the simulated annealing algorithm and chemotaxis algorithm.

Yang *et al.* [20] have introduced a hybrid chaos optimization algorithm with artificial emotion (HCOAAE) to avoid a premature convergence and increase its exploration in a high-dimension space. The main purpose of HCOAAE is to mimic the decision-making behavior process of humans in choosing the parameters of HCOAAE and decide whether to change the current search strategy or not (in the next iteration). Their experimental results over 13 benchmark functions show that HCOAAE significantly outperforms the state-of-the-art methods in terms of the smooth convergence behavior, computational complexity, and numerical stability. Dong *et al.* [21] have propose an evolutionary circle detection method based on a novel chaotic hybrid algorithm (CHA). Their method combines the strengths of PSO, GA, and chaotic dynamics. CHA adopts the standard velocity and position update rules of PSOs with the ideas of selection, cross-over, and mutation from GA with the opposition-based learning for population initialization. They demonstrated the effectiveness of CHA in the circle detection problems.

Alatas *et al.* [22] have proposed an improved PSO method equipped with chaotic maps for its parameter adaptation. After a few iterations, a new set of chaotic numbers are generated to update the parameters of PSO. They implemented twelve chaos-embedded PSO methods and used eight different chaotic maps to generate random numbers and applied them to a benchmark of complex functions. Inserting a high degree of randomness improved their exploration as well as avoiding the premature convergence.

Gao *et al.* [23] have introduced a new hybrid PSO, which incorporates the Henon map mutation operation (HPSO) to enhance the exploration ability of PSO. Their new mutation strategy divides the mutation operator into the global and local mutation operators, enabling particles to move in the search space with different step sizes. Their comparison results imply the superiority of HPSO over other hybrid PSO algorithms over all the 16 complex functions. Jia *et al.* [24] have introduced a new mimetic PSO (CGPSO) by equipping the standard PSO with a chaotic function and Gaussian local search procedure. Using a chaotic local search enables it to widely explore as well as avoid the premature convergence. In addition, the solutions are refined through Gaussian optimization. Their results over thirteen benchmark functions show that CGPSO is more effective, faster to converge,

and less sensitive to the dimensions of the search space in comparison with the other chaotic PSO variants.

The contribution of this paper is to insert a chaotic term in the updating formulas of AIW-PSO as well as initialization of the particles in order to increase its exploration property. In the early iterations of the proposed scheme, the perturbation rate is high, and the particles are moved according to the quasi-random values that the chaotic functions generate. As the iteration number is increased, the threshold (perturbation rate) is decreased, and the movement of particles obeys the AIW-PSO algorithm more. In other words, at first, the random movements lead to explore and scatter the particles in the search space, and little by little (by increasing the number of iterations), the exploitation capability of the algorithm is increased. Preserving both the exploration and exploitation capabilities is impossible but in this manner, we increased the diversity (exploration) at the first iterations of the algorithm and then the particles were exploited around a suitable local optima, though in a few iterations the particles might move randomly.

The rest of this paper is structured as what follows. Section 2 proposes the chaotic-based PSO method. Sections 3 expresses the experimental results over the benchmark of complex functions, and their pros and cons are discussed. Finally, the paper is concluded in Section 4.

2. Materials and methods

In this section, the benchmark of complex functions [25-27] is expressed to assess the proposed methods in finding the minimum values for each function. Next, the standard PSO and chaotic-based PSO are explained in detail.

2.1. Benchmark functions

In order to assess the proposed methods, a known benchmark of complex functions [25-27] was employed, which are described in tables 1, 2, and 3. Each one of these functions has several local minima. Moreover, the global minimum (grand truth) as well as the search range for each function is provided.

2.2. Chaos theory

Chaotic systems [28] are rule-based and deterministic systems that are mainly characterized by time-varying or non-linear differential equations. The output of these systems is highly sensitive to their initial conditions in a way that by changing a very small change in their values, the behavior of their output signal is significantly changed. Their output signals are irregular and behave like noise, and therefore, these outputs are mainly used as a random signal generator. This randomness can be inserted in the movement of particles in PSO for diversing the population and avoiding the premature convergence. In addition, these irregular signals are repeatedly used for the initialization of a diverse population [22]. What follows is the description of three known chaotic systems that have been formerly used for improving different evolutionary algorithms [29].

2.2.1. Lorenz system

The Lorenz system [30] is one of the well-known chaotic systems that are originally derived from a model of the earth's atmospheric convection flows of heating and cooling from below and above, respectively. This system is described by three coupling differential equations:

$$\begin{aligned} \frac{dx}{dt} &= \delta(y-x) \\ \frac{dy}{dt} &= x(\rho-z)-y \\ \frac{dz}{dt} &= xy-\beta z \end{aligned} \qquad (1)$$

where, x, y and z are the state variables of the system, t is the time, and σ, β, ρ are the system parameters. For the values of $\sigma = 10,\ \beta = \frac{8}{3}$ *and* $\rho = 28$ the Lorenz system exhibits a chaotic behavior, as shown in figure 1.

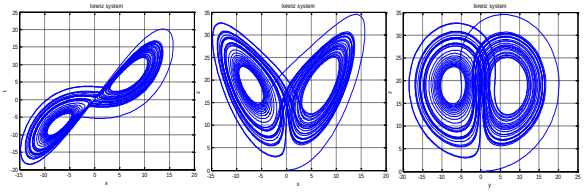


Figure 1. Chaotic behavior of the state variables of the Lorenz system.

2.2.2. Henon map

The Henon map [31] is a simplified version of the Poincare map of the Lorenz system. The Henon map equations are described below:

$$\begin{cases} x_{n+1} = 1 + y_n - a x_n^2 \\ y_{n+1} = b x_n \end{cases} \qquad (2)$$

where, x and y are the state variables of this system, and the parameters a and b can control the behavior of the system. For the values of $a = 1.4$ and $b = 0.3$, the Henon map shows a chaotic behavior, as shown in figure 2.

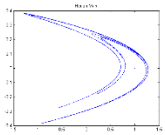


Figure 2. Chaotic behavior of the Henon map.

Table 1. Unimodal test functions (*D* = 30 in this work).

Functions	Range
$f_1(x)=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$
$f_2(x)=\sum_{i=1}^{D} x_i^2$	$[-10,190]^D$
$f_3(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$[-10,10]^D$
$f_4(x)=\sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2$	$[-100,100]^D$
$f_5(x)=(\sum_{i=1}^{n} ix_i^4)+rand[0,1]$	$[-1.28,1.28]^D$
$f_6(x)=\sum_{i=1}^{n-1}[100(x_i^2-x_{i+1})^2+(x_i-1)^2]$	$[-5,10]^D$
$f_7(x)=\sum_{i=1}^{D}(\lfloor x_i+\frac{1}{2}\rfloor)^2$	$[-10,10]^D$

Table 2. Multimodal test functions (*D* = 30 in this work).

Functions	Range
$f_8(x)=0.1sin^2(3\pi x_1)+\sum_{i=1}^{n}((x_i-1)^2+sin^2(3\pi x_1+1))^2+((x_D-1)^2(1+sin^2(2\pi x_n)))+\sum_{i=1}^{n}u(x_i,5,100,4)$	$[-50,50]^D$
$f_9(x)=\sum_{i=1}^{D}-x_i sin(\sqrt{\lvert x_i\rvert})$	$[-500,500]^D$
$f_{10}(x)=\sum_{i=1}^{n}(x_i^2-10cos(2\pi x_i))+10)$	$[-512,512]^D$
$f_{11}(x)=\sum_{i=1}^{n} y_i^2-10cos(2\pi y_i))+10,\ y_i=\begin{cases} x_i & \lvert x_i\rvert<0.5 \\ \frac{round(2x_i)}{2} & \lvert x_i\rvert\geq 0.5 \end{cases}$	$[-512,512]^D$
$f_{12}(x)=-20exp(-0.2\sqrt{\frac{1}{30}\sum_1^n x_i^2})-exp(\frac{1}{D}\sum_{i=1}^{D}cos(2\pi x_i))+20+e$	$[-32,32]^D$
$f_{13}(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}cos(\frac{x_i}{\sqrt{i}})+1$	$[-600,600]^D$
$f_{14}(x)=sin^2(\pi y_1)+\sum_{i=1}^{n}((y_i-1)^2+10sin^2(\pi y_1+1))^2+((y_D-1)^2(1+sin^2(2\pi y_D)),\ y_i=1+\frac{x_i-1}{4}$	$[-10,10]^D$
$f_{15}(x)=(\sum_{i=1}^{D}icos(i+1)x_1+i))(\sum_{i=1}^{5}icos(i+1)x_2+i))$	$[-10,10]^D$

Table 3. Multimodal test functions with fix dimension.

Functions	Range
$f_{16}(x)=(x_2-\frac{5.1}{4\pi^2}^2 x_1^2+\frac{5}{\pi}x_1-6)^2+10(1-\frac{1}{8\pi})cos(x_1)+10$	$-5\leq x_1\leq 10$ $0\leq x_2\leq 15$
$f_{17}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	$[-5,5]^4$
$f_{18}(x)=4x_1^2-2.1x_1^4+\frac{1}{3x_1^6}+x_1x_2-4x_2^2+4x_2^4$	$[-5,5]^2$
$f_{19}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]*[30+(2x_1-3x_2)^2*$ $(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	$[-5,5]^2$
$f_{20}(x)=-\sum_{i=1}^{4}c_i exp(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2)$	$[0,1]^3$
$f_{21}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	$[0,10]^4$

2.2.3. Tent map

The Tent map [32] is the simplest kind of 1D chaotic dynamic mapping, which is described in Eq. (3):

$$x_{n+1} = f_{\mu}^{(x_n)} = \begin{cases} \mu(x_n) & for\ x_n < \frac{1}{2} \\ \mu(1-x_n) & for\ x_n \geq \frac{1}{2} \end{cases} \quad (3)$$

where, x is the state variable of the system, and for $1 < \mu < 2$, the system is in a chaotic state. Figure 3 shows the behavior of the chaotic Tent map.

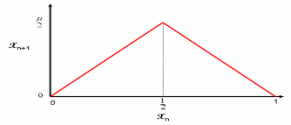


Figure 3. Chaotic behavior of the Tent map.

2.3. Particle swarm optimization (PSO)

At the beginning, PSO creates a population of candidate solutions, which are so-called particles. They are randomly generated, and each particle of this population has a potential for being a proper solution after some epochs. At each iteration, the particles are optimized under two different criteria; $gbest$ and $pbest$, which, respectively, assess the global and local fitness of each particle. The i^{th} particle is denoted as $X_i = (x_{i1}, x_{i2}, ..., x_{is})$, where s is the dimension on the particles. The velocity and position of each particle are updated according to the following relations:

$$v_{id} = w * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \quad (4)$$

$$x_{id} = x_{id} + v_{id} \quad (5)$$

where, $d = 1, 2, ..., s$ and w is the inertia weight. The two acceleration parameters c_1 and c_2 represent the weight of the stochastic acceleration terms that pull each particle toward the $Pbest(p_{id})$ and $gbest(p_{gd})$ positions. $Rand()$ and $rand()$ are two random functions in the range of $[0,1]$. The velocity of each particle (v) is limited between $w_{\min}$ and $w_{\max}$ that are defined by the user as the input parameters that determine the step size of each particle through the solution space.

2.4. Chaotic-based adaptive inertia weight PSO (AIW-PSO)

AIW-PSO [33] uses the success rate of the whole particles as a feedback parameter to control the particles' movement in the search space. The inertia weight (w) is one of the PSO parameters, which makes a balance between the exploration and exploitation of the particles. AIW-PSO is required to determine the situation (success rate in percentage) of the swarm at each epoch. A high success rate indicates that the particles have converged to a point that is far from the optimum point, and the entire swarm slowly move toward the optimum. In contrast, a low success rate shows that the particles are oscillating around the optimum without a significant improvement. The success rate of the swarm is calculated as follows:

$$P_s(t) = \frac{\sum_{id=1}^{n} s(id,t)}{n} \quad (6)$$

where, n is the number of particles, $s(id,t)$ is the number of particles that have had an improvement in their fitness in the last iteration. Finally, the inertia weight is updated as follows:

$$w(t) = (w_{\max} - w_{\min}) P_s(t) + w_{\min} \quad (7)$$

where, the range of the inertia weight $([w_{\min}, w_{\max}])$ is randomly selected within the range of $[0,1]$.

In order to find the extremum points of complex non-linear problems, similar to PSO, AIW-PSO may lead to premature convergence [34, 35]. In order to overcome this drawback, the three mentioned chaos systems are employed to increase the diversity among the initial population.

In addition, the perturbation stage is added, which permits more exploration, traverse the search space sufficiently, and decrease the chance of premature convergence. In this work, the velocity of particles is perturbed as follows:

$$v_{id} = w * v_{id} + c_1 * rand() * NewP + c_2 * rand() * NewG \quad (8)$$

$$NewP = \begin{cases} (p_{id} - x_{id}) & rand() \geq r_p \\ rand() & otherwise \end{cases} \quad (9)$$

$$NewG = \begin{cases} (p_{gd} - x_{id}) & rand() \geq r_p \\ rand() & otherwise \end{cases} \quad (10)$$

where, the perturbation rate, r_p , is decreased during the iterations. Figure 4 describes the pseudo-code of the chaotic-based AIW-PSO.

3. Experimental results and discussion

In this work, the proposed chaotic AIW-PSO along with AIW-PSO, chaotic PSO, and GA are applied to 21 complex functions [36] (described in tables 1, 2, and 3). The proper values of parameters are selected through the cross-validation phase. Table 4 illustrates the selected values of the parameters for the AIW-PSO and GA methods. The results of applying the compared methods to the functions are presented in table 5. In order to evaluate the chaotic AIW-PSO, three chaotic systems are executed to generate the random sequences for initializing the population of the standard and the chaotic-based AIW-PSO methods.

Table 4. Values of the parameters for the chaotic AIW-PSO and GA.

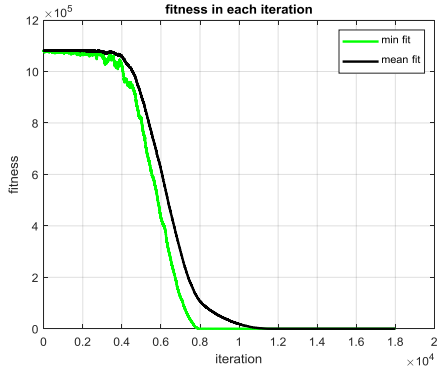
AIW-PSO		GA	
Parameter	**Value**	**Parameter**	**Value**
Particles	20	Population size	20
Maximum iterations	20000	Maximum iterations	20000
V_{min}	-0.15	Crossover rate	0.9
V_{max}	0.15	Mutation rate	0.1
c_1	2		
c_2	2		

The results obtained show that the chaotic-based AIW-PSO (with Lorenz system) obtains the best results among the other selected optimization algorithms. In the Lorenz chaotic system, the perturbation rate is initially set to 0.8 but in the Henon and Tent maps, this parameter is initially set to 0.3. Nonetheless, in all cases, this rate is linearly decreased through successive iterations. For some functions such as $f_{15}(x)$, we have increased the perturbation rate for improving the accuracy of the optimization work. Our finding shows that this function requires more exploration in comparison with the other functions.

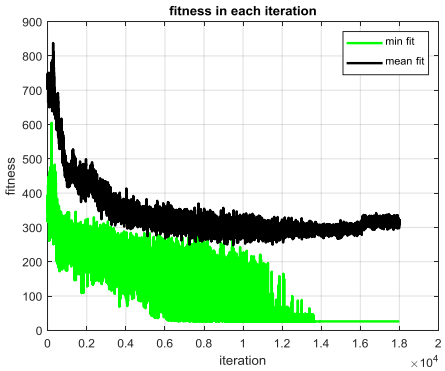
$$
\begin{aligned}
&Initialize\ Particle\{x_{id}, v_{id}\}\ based\ on\ chaos\ theory\\
&while(Iter < MaxGen \&\& Gbest < M\ ax\ fit)\{\\
&\quad for(every\ particle\ i)\{\\
&\quad\quad Fitness(i) = statistical\_Evaluation(i);\\
&\quad\quad if\ (fitness(i) > pbest(i))\{\\
&\quad\quad\quad pbest(i) = fitness(i);\\
&\quad\quad\quad p_{id} = x_{id}\\
&\quad\quad\}\\
&\quad\quad if\ (fitness(i) Gbest)\{\\
&\quad\quad\quad Gbest = fitness(i);\\
&\quad\quad\quad gbest = i;\\
&\quad\quad\}\\
&\quad\}\\
&\quad for(every\ particle\ i)\{\\
&\quad\quad for(every\ particle\ d)\{\\
&\quad\quad\quad NewP = \begin{cases}(p_{id} - x_{id}) & rand() \geq r_p\\ rand() & otherwise\end{cases}\\
&\quad\quad\quad NewG = \begin{cases}(p_{gd} - x_{id}) & rand() \geq r_p\\ rand() & otherwise\end{cases}\\
&\quad\quad\quad v_{id} = w * v_{id} + c_1 * rand() * NewP +\\
&\quad\quad\quad\quad c_2 * rand() * NewG\\
&\quad\quad\quad x_{id} = x_{id} + v_{id}\\
&\quad\quad\}\\
&\quad\}\\
&\quad Iter = iter + 1;\\
&\}
\end{aligned}
$$

Figure 4. Pseudo-code of the chaotic-based PSO algorithm.

Our statistical evaluation using the Students' T-test show that there is a significant difference between the results of the chaotic-based AIW-PSO (with Lorenz system) and the other optimization algorithms in some of the functions. Figures 5, 6, and 7 show the convergence of chaotic-based AIW-PSO for three sample functions.

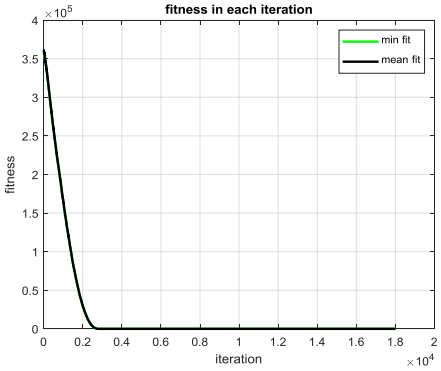


(a)

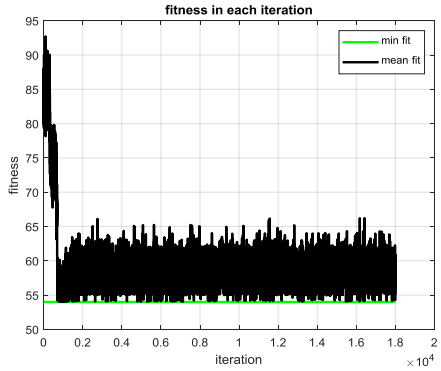


(b)

Figure 5. Convergence of the chaotic-based AIW-PSO with Lorenz system for two sample functions (a) f_2 (b) f_{11}.



(a)

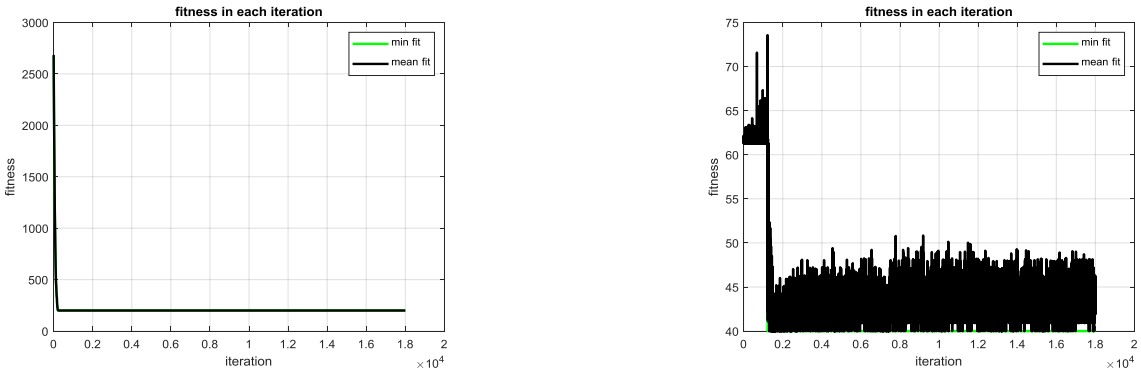


(b)

Figure 6. Convergence of the chaotic-based AIW-PSO with Henon map for two sample functions (a) f2 (b) f11.

Table 5. The mean and Std of the best solution of the GA, AIW-PSO, chaotic-based PSO, and chaotic-based AIW-PSO methods over the complex functions.

Function	GA	AIW-PSO	Chaotic-based PSO			Chaotic-based AIW-PSO		
			Lorenz	Henon	Tent	Lorenz	Henon	Tent
$f_1(x)$	0.0± 0.04	0.00 ± 0.00	0.16 ± 0.01	0.15 ± 0.03	0.15 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_2(x)$	0.0± 0.0	1.29e3 ± 523.76	0.17 ± 0.02	1.01e3 ± 795.01	760.10 ± 560.13	0.00 ± 0.00	0.00 ± 0.00	270.00 ± 188.86
$f_3(x)$	-3.43 ± 2.44	0.00 ± 0.00	1.73 ± 0.12	1.69 ± 0.17	1.78 ± 0.09	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_4(x)$	-222.98± 21.38	0.00 ± 0.00	2.50 ± 0.57	2.29 ± 0.49	2.46 ± 0.34	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_5(x)$	3.04± 0.91	0.05 ± 0.04	0.15 ± 0.04	0.11 ± 0.03	0.15 ± 0.04	0.05 ± 0.04	0.03 ± 0.02	0.05 ± 0.06
$f_6(x)$	28.88 ± 34.35	0.83 ± 1.67	41.17 ± 3.36	52.62 ± 20.16	51.01 ± 19.81	3.85 ± 2.52	2.38 ± 2.72	4.42 ± 2.43
$f_7(x)$	23.00 ± 5.81	18.90 ± 6.57	0.00 ± 0.00	31.90 ± 9.77	15.20 ± 8.23	0.00 ± 0.00	36.90 ± 13.92	24.30 ± 13.77
$f_8(x)$	0.00 ± 0.00	30.40 ± 5.50	0.03 ± 0.00	26.99 ± 6.63	50.09 ± 10.51	0.01 ± 0.01	32.22 ± 2.80	47.31 ± 6.32
$f_9(x)$	-483.68± 66.12	-377.10 ± 56.34	-418.98 ± 0.00	-418.98 ± 0.00	-418.98 ± 0.00	-418.98 ± 0.00	-418.98 ± 0.00	-418.98 ± 0.00
$f_{10}(x)$	-3.25e3± 17.64	0.00 ± 0.00	0.00 ± 0.01	0.00 ± 0.01	0.03 ± 0.04	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_{11}(x)$	44.27 ± 14.42	39.20 ± 13.71	45.85 ± 12.59	74.61 ± 22.91	58.83 ± 11.48	29.80 ± 0.42	76.92 ± 19.81	39.70 ± 9.18
$f_{12}(x)$	2.81 ± 0.74	9.08 ± 1.28	2.76 ± 0.44	18.11 ± 0.19	9.24 ± 1.81	2.66 ± 0.24	12.65 ± 0.24	8.70 ± 1.30
$f_{13}(x)$	2.42 ± 4.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_{14}(x)$	27.33 ± 25.42	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_{15}(x)$	-3.02e3± 1.47e3	-186.73 ± 0.00	11.18 ± 0.00	5.32e15 ± 0.00	5.32e15 ± 0.00	11.18 ± 0.00	5.32e15 ± 0.00	5.32e15 ± 0.00
$f_{16}(x)$	-0.66 ± 5.11	0.39 ± 0.00	0.85 ± 0.49	0.59 ± 0.21	0.75 ± 0.33	0.40 ± 0.00	0.40 ± 0.00	0.40 ± 0.00
$f_{17}(x)$	0.00 ± 0.00	0.01 ± 0.01	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.01	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
$f_{18}(x)$	-1.03 ± 0.00	-1.03 ± 0.00	-1.03 ± 0.00	-1.03 ± 0.00	-1.02 ± 0.01	-1.03 ± 0.00	-1.03 ± 0.00	-1.03 ± 0.00
$f_{19}(x)$	3.00 ± 0.00	3.00 ± 0.00	22.05 ± 13.12	85.57 ± 1.28	3.27 ± 0.23	30.00 ± 0.00	84.00 ± 0.00	11.10 ± 25.61
$f_{20}(x)$	-3.88 ± 0.00	-3.88 ± 0.00	-3.84 ± 0.04	-3.86± 0.02	-3.85 ± 0.02	-3.88 ± 0.00	-3.88 ± 0.00	-3.88 ± 0.00
$f_{21}(x)$	-9.41 ± 2.36	-6.38 ± 3.39	-9.22 ± 0.39	-5.01 ± 0.06	-4.14 ± 2.40	-10.15 ± 0.00	-5.10 ± 0.00	-4.94 ± 2.66



(a) (b)

Figure 7. Convergence of the chaotic-based AIW-PSO with Tent map for two sample functions (a) f2 (b) f11.

With a hard look over figures 5-7, we can come into this conclusion that the Lorenz system can generate sequences with more randomness, which enables the proposed chaotic-based AIW-PSO to fall less into the local extremums (premature convergence) because this randomness increases its exploration and exploitation capabilities. In addition, AIW-PSO encodes the experience of all particles into the inertia weight in each pace such that each particle is being awarded about the status of the whole particles and then moves chaotically in each epoch accordingly. Hence, the proposed manner benefits from a high degree of randomness to avoid premature convergence while the particles share their status. This leads to a fast and stable convergence, where the error is smoothly decreased through successive iterations. A closer look at the convergence curves of the compared methods over the benchmark functions provides more insight into the behavior of the algorithms. As one can see in figures 5-7, the learning curves of the proposed methods on the optimization benchmark imply a fast and smooth convergence behavior compared to the counterparts.

In this work, we adopted two measures [37] for determining the population diversity of different PSO algorithms: population fitness standard deviation and population position standard deviation.

Population fitness standard deviation: If the particles of a population $S=(X_1,...,X_i,...,X_N)$ at generation t get their fitness value $(f_1(t),...,f_i(t),...,f_N(t))$, the population fitness standard deviation of different PSO algorithms is defined as:

$$STD_{fitness(t)}=\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(f_i(t)-\bar{f}(t))^2} \quad (11)$$

where, $\bar{f}(t)=\sum_{i=1}^{N}f_i(t)$.

Population position standard deviation: If the particles of a population $S=(X_1,...,X_i,...,X_N)$ at generation t get their positions $X_1(t),...,X_i(t),...,X_N(t)$, the population position standard deviation for generation t can be computed as:

$$stdev^{(j)}(t)=\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(X_{ij}(t)-\bar{X}^j(t))^2} \quad (12)$$

where, $X_i(t)=X_{i1}(t),X_{i2}(t),...,X_{iD}(t)$, $\bar{X}(t)$ can be expressed as $\bar{X}^{(1)}(t)\bar{X}^{(2)}(t)...\bar{X}^{(D)}(t)$ and $\bar{X}^{(j)}(t)=\frac{1}{N}\sum_{i=1}^{N}(X_{ij}(t)$, figure 8 shows the diversity of populations for a randomly selected function. As shown in this figure, the chaotic-based PSO with Lorenz system has a more diversity (more exploration) in comparison with the other algorithms.

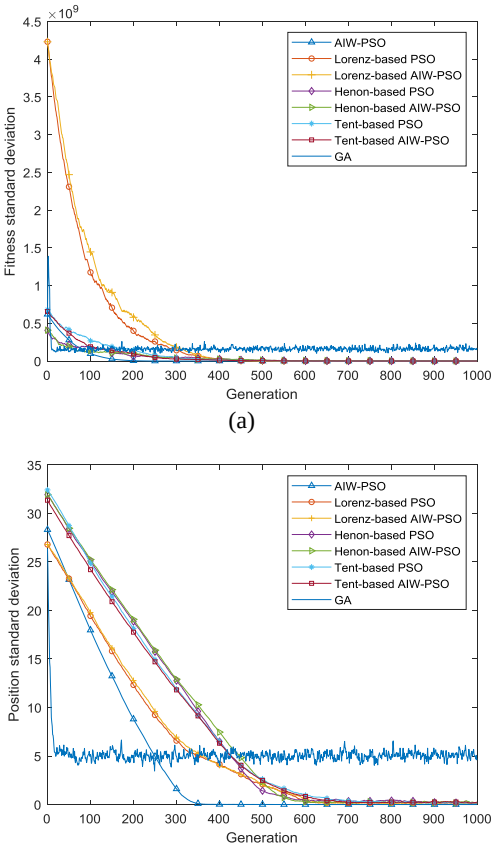


Figure 8. (a) Population fitness standard deviation (b) Population position standard deviation for different algorithms.

The chaotic operators have been applied in order to improve the performance of GA, ACO, and also other evolutionary algorithms. Nonetheless, GA has highly effective operators (cross-over and mutation) to improve its exploration capacity, and adding the chaotic operator does not highly affect its performance. GA suffers from the exploitation ability that the chaotic operators cannot help GA in this regard. Furthermore, ACO obeys a probabilistic function when an ant wants to choose a path. Similar to PSO, the chaotic operators can help both GA and ACO to diversitize their initial population but to increase the randomness of GA and ACO operators, the chaotic functions cannot help them that much.

The PSO algorithm is naturally a continuous optimization algorithm; consequently, the proposed methods provided a significantly superior

result in terms of finding extremum points over the continuous functions in comparison with the GA and ant colony optimization (ACO) algorithms. Therefore, GA and ACO are naturally designed for discrete optimization problems. In the employed benchmark, there are two discrete functions, and as we have expected, the PSO versions cannot outperform GA on these cases, though the proposed chaotic AIW-PSO provides better results than that the other implemented PSO versions (see Table 5).

4.Conclusion

In this paper, some chaotic-based PSO versions equipped with the inertia weight strategy are suggested as strong optimizers for continuous optimization problems. The inertia weight depends on the size of the particles and integrate the experience of all particles as an auxiliary clue to better guide the particles toward the extremum points. In this strategy, each particle is awarded the state of the particles in the search space by this inertia weight (as a feedback parameter). In order to avoid the premature convergence, the three known chaotic systems (Lorenz system, Tent map, and Henon map) are implemented to generate quasi-random sequences to be inserted into the updating position of particles. Among the chaotic systems employed, sequences that are generated by the Lorenz system have led to a better optimization performance as well as producing a smoother convergence behavior through a lesser number of epochs, compared with AIW-PSO. The suggested chaotic AIW-PSO can be considered as an alternative scheme to solve a wide range of multi-dimensional complex optimization problems, especially for continuous problems.

References

[1] Glover, F.W. & Laguna, M. (1997). Tabu Search. New York: Springer-Verlag.

[2] Chibante, R. (2010). Simulated annealing, theory with applications. IntechOpen.

[3] Rezaee, N. & Momeni, H. (2020). A hybrid meta-heuristic approach to cope with state space explosion in model checking technique for deadlock freeness. Journal of AI and Data Mining, vol. 8, no. 2, pp. 189-199.

[4] Simon, D. (2013). Evolutionary optimization algorithms, United States: Wiley.

[5] Affenzeller, M., Winkler, S., Wagner, S. & Beham, A. (2009). Genetic algorithms and genetic programming: modern concepts and practical applications. New York: Chapman & Hall.

[6] Eberhart, R.C. & Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. IEEE International Conference on Evolutionary Computation, Seoul, South Korea, 2001.

[7] Clerc, M. (2006). Particle swarm optimization. Wiley-ISTE Ltd.

[8] Guo, Y., Wu, Z., Wang, Y. & Wang Y. (2016). Extended particle swarm optimization method for folding protein on triangular lattice. IET Systems Biology, vol. 10, no. 1, pp. 30-33.

[9] Guo, Y., Tao, F., Wu, Z. & Wang, Y. (2017). Hybrid method to solve HP model on 3D lattice and to probe protein stability upon amino acid mutations. BMC Systems Biology, vol. 11, no. 4, pp. 93-106.

[10] Boostani, R. & Sabeti, M. (2017). Can evolutionary-based brain map use as a complementary diagnostic tool with fMRI, CT and PET for schizophrenic patients? Journal of Biomedical Physics and Engineering, vol. 7, no. 2, pp. 169-180.

[11] Boostani, R. & Sabeti, M. (2018). Optimizing brain map for the diagnosis of schizophrenia. International Journal of Biomedical Engineering and Technology, vol. 28, no. 2, pp. 105-119.

[12] Zelinka, I., Celikovsky, S., Richter, H. & Chen G. (2010). Evolutionary algorithms and chaotic systems (Studies in Computational Intelligence), Berlin: Springer-Verlag.

[13] Javidi, M. & Hosseinpourfard, R. (2015). Chaos genetic algorithm instead genetic algorithm. International Arab Journal of Information Technology, vol. 12, no. 2, pp. 163-168.

[14] Hosseinpourfard, R. & Javidi, M.M. (2015). Chaotic PSO using the Lorenz system: an efficient approach for optimizing nonlinear problems. Çankaya University Journal of Science and Engineering, vol. 12, no. 1, pp. 40-59.

[15] Snaselova, P. & Zboril, F. (2015). Genetic algorithm using theory of chaos. Procedia Computer Science, vol. 51, pp. 316-325.

[16] Abdullah, A., Enayatifa, R. & Lee M. (2012). A hybrid genetic algorithm and chaotic function model for image encryption. Journal of Electronics and Communication, vol. 66, no. 10, pp. 806-816.

[17] Wang, Y. & Yao, M. (2009). A new hybrid genetic algorithm based on chaos and PSO. IEEE International Conference on Intelligent Computing and Intelligent Systems, Shanghai, China, 2009.

[18] Yang, C.H., Tsai, S.H., Chuang, L.I. & Yang, C.H. (2012). An improved particle swarm optimization with double-bottom chaotic maps for numerical optimization. Applied Mathematics and computation, vol. 219, no. 1, pp. 260-279.

[19] Li, B. & Jiang, W. (1998). Optimizing complex functions by chaos search. Journal of Cybernetics and Systems, vol. 29, no. 4, pp. 409-419.

[20] Yang, Y., Wang, Y., Yuan, X. & Yin, F. (2012). Hybrid chaos optimization algorithm with artificial emotion. Applied Mathematics and Computation, vol. 218, no. 11, pp. 6585-6611.

[21] Dong, N., Wu, C.H., Ip W.H., Chen, Z.Q., Chan, C.H. & Yang, K.L. (2012). An opposition –based chaotic GA/PSO hybrid algorithm and its application in circle detection. Computers and Mathematics with Application, vol. 64, no. 6, pp. 1886-1902.

[22] Alatas, B., Akin, E. & Ozer, A. (2009). Chaos embedded particle swarm optimization algorithms. Chaos Soliton & Fractals, vol. 40, no. 4, pp. 1715-1734.

[23] Gao, H. & Xu W. (2011). Particle swarm algorithm with hybrid mutation strategy. Applied Soft Computing, vol. 11, no. 8, pp. 5129-5142.

[24] Jia, D., Zheng, G., Qu B. & Khan, M.K. (2011). A hybrid particle swarm optimization algorithm for high-dimensional problems. Computers & Industrial Engineering, vol. 61, no. 4, pp. 1117-1122.

[25] Digalakis, J.G. & Margaritis, K.G. (2001). On benchmarking functions for genetic algorithms. International Journal of Computer Mathematics, vol. 77, no. 4, pp. 481-506.

[26] Rajput, S., Parashar, M. & Dubey, HM. (2016). Optimization of benchmark functions and practical problems using crow search algorithm. International Conference on Eco-friendly Computing and Communication Systems, Bhopal, India, 2016.

[27] Cheng, R., Li, M., Tian, Y., Zhang, X., Yang, S., Jin, Y. & Yao, X. (2017). A benchmark test suite for evolutionary many-objective optimization. Complex & Intelligent Systems, vol. 3, pp. 67-81.

[28] Gleick J., (1987). Chaos: making a new science. New York: Viking Press.

[29] El-Shorbagy, M.A., Mousa, A.A. & Nasr, S.M. (2016). A chaos-based evolutionary algorithm for general nonlinear programming problems. Chaos, Solitons & Fractals, vol. 85, pp. 8-21.

[30] Ghys, E. (2013). The Lorenz attractor, a paradigm for chaos. Chaos, pp. 1-54.

[31] Cai, M. (2015). Complex dynamics in generalized Henon map. Discrete Dynamics in Nature and Society, vol. 2015, Article ID 270604.

[32] Li, C., Luo, G., Qin, K. & Li, C. (2017). An image encryption scheme based on chaotic tent map. Nonlinear Dynamics, vol. 87, pp. 127- 133.

[33] Nickabadi, A., Ebadzadeh, M.M. & Safabakhsh, R. (2011). A novel particle swarm optimization algorithm with adaptive inertia weight. Applied Soft Computing, vol. 11, no. 4, pp. 3658-3670.

[34] Dong, N., Fang, X. & Wu, A. (2016). A novel chaotic particle swarm optimization algorithm for parking space guidance. Mathematical Problems in Engineering, vol. 2016, Article ID 5126808.

[35] Xu, X., Rong, H., Trovati, M., Liptrott, M. & Bessis N. (2018). CS-PSO: chaotic particle swarm optimization algorithm for solving combinatorial optimization problems, Soft Computing, vol. 22, pp. 783-795.

[36] Rashedi, E., Nezamabadi-pour, H. & Saryazdi, S. (2009). GSA: a gravitational search algorithm. Information Sciences, vol. 179, no. 13, pp. 2232–2248.

[37] Ni, Q. & Deng, J. (2014). Analysis of population diversity of dynamic probabilistic particle swarm optimization algorithms. Mathematical Problems in Engineering, vol. 2014, Article ID 762015.

نشریه هوش مصنوعی و داده کاوی

استفاده از الگوریتم بهینه‌سازی ازدحام ذرات آشوبگون با اینرسی تطبیق شونده در مسائل بهینه‌سازی

ندا مبارکی [۱]، رضا بوستانی [۲،*] و ملیحه ثابتی [۳]

[۱] گروه مهندسی کامپیوتر، موسسه آموزش عالی آپادانا، شیراز، ایران.

[۲] گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشکده مهندسی برق و کامپیوتر، دانشگاه شیراز، شیراز، ایران.

[۳] گروه مهندسی کامپیوتر، واحد تهران شمال، دانشگاه آزاد اسلامی، تهران، ایران.

ارسال ۲۰۱۹/۰۶/۱۸؛ بازنگری ۲۰۲۰/۰۲/۲۲؛ پذیرش ۲۰۲۰/۰۴/۱۵

چکیده:

در میان انواع الگوریتم‌های جستجوی فرا ابتکاری مبتنی بر جمعیت، الگوریتم بهینه‌سازی ازدحام ذرات (PSO) با وزن اینرسی تطبیق شونده (AIW) را می‌توان بعنوان ابزار بهینه سازی کارایی در نظر گرفت که تجربه کل جمعیت را با حرکت ذرات جابجا می‌کند. اگرچه این الگوریتم توانایی بالایی دارد اما این الگوریتم نمی‌تواند فضای جستجو را به صورت جامع جستجو کند و ممکن است پس از تعداد محدودی تکرار در حداقل محلی گرفتار شود. در این مطالعه برای افزایش تنوع و همچنین افزایش توانایی اکتشاف ذرات در AIW-PSO ، از سه تابع آشوبگون برای مقدار دهی اولیه ذرات و سپس اعمال اغتشاش در ذرات در حین جستجو برای جلوگیری از همگرایی زودرس بویژه در توابع پیچیده غیرخطی استفاده می‌شود. برای ارزیابی روش پیشنهادی، از توابع بهینه سازی شناخته شده‌ای شامل توابع پیچیده غیرخطی استفاده شده و نتایج آن با نتایج الگوریتم استاندارد PSO، الگوریتم AIW-PSO و الگوریتم ژنتیک (GA) مقایسه شده است. نتایج تجربی این مطالعه بر روی بیش از ۲۱ تابع نشان از برتری روش پیشنهادی AIW-PSO نسبت به همتایان دارد که تاثیر مثبت افزایش تصادفی بودن ذرات در الگوریتم AIW-PSO را تأیید می‌کند. تحلیل رفتار ذرات نشان می‌دهد که روش پیشنهادی می‌تواند بدون مشکل همگرایی زودرس، مینیمم مناسب را پیدا کند.

کلمات کلیدی: الگوریتم PSO-AIW، تصادفی، تئوری آشوبگون، تجربه ذرات، نرخ همگرایی.