BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **15**, No 6
Special Issue on Logistics, Informatics and Service Science

Sofia • 2015 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2015-0068

Particle Swarm Optimization Algorithm with Adaptive Chaos Perturbation

Dong Yong[1], *Wu Chuansheng*[2], *Guo Haimin*[3]

[1]*School of Information and Mathematics of Yangtze University, Hubei Jingzhou, 434023 China; Institute of Petroleum Engineering Technology, Zhongyuan Bureau of Petroleum Exploration, Puyang Henan,457001 China*
[2]*Mechanical Postdoctoral Station, Wuhan University of Technology, Wuhan Hubei, 430070 China*
[3]*Key Laboratory of Exploration Technologies for Oil and Gas Resources, Ministry of Education, Yangtze University, Hubei Wuhan, 430100 China*
Emails: dongyong80@126.com lzywcs@whut.edu.cn ghm3819@263.net

Abstract: *Most of the existing chaotic particle swarm optimization algorithms use logistic chaotic mapping. However, the chaotic sequence which is generated by the logistic chaotic mapping is not uniform enough. As a solution to this defect, this paper introduces the Anderson chaotic mapping to the chaotic particle swarm optimization, using it to initialize the position and velocity of the particle swarm. It self-adaptively controls the portion of particles to undergo chaos update through a change of the fitness variance. The numerical simulation results show that the convergence and global searching capability of the modified algorithm have been improved with the introduction of this mapping and it can efficiently avoid premature convergence.*

Keywords: *Chaos, particle swarm optimization, fitness variance, convergence rate.*

1. Introduction

The Particle Swarm Optimization (PSO) is an evolution algorithm based on swarm intelligence. Kennedy and Eberhart [1], and Clerc and Kennedy [2] have investigated PSO in 1995. Their study involved the simulation of the

migration and assembling behaviour of predatory birds while they are on the hunt. The PSO has many advantages, such as simple parameters, being easily realized, parallel processing capability, and robust characteristics. PSO can converge to the global optimal solution with a high degree of probability. Due to this most valuable advantage it is widely utilized. It has successfully been used in the field of image processing as investigated by Chatterjee and Siarry [3], the economic optimization problem as presented in Vaisakh et al. [4], the expert system as in Babu et al. [5], the industrial control and application as given in Niknam [6], Lin et al. [7], Piancastelli et al. [8], Wang [9] and Wang et al. [10], path planning as in Kundra and Sood [11]. Premature convergence occurs with PSO particularly in complex high-dimensional and multimodal search problems. Solving this problem requires a method which combines PSO with the chaotic system using the pseudo-randomness and ergodicity of the chaotic system to perturb the particle to advance the probability and velocity of the global convergence as investigated by Liu and Lin [12].

The most widely applied PSO which combined chaos mapping used a logistic map to generate chaos sequence as investigated by Liu and Lin [12], Yang [13], and Lu and Liu [14]. However, the random sequence generated by logistic chaotic mapping has poor uniformity, thus affecting the convergence of the algorithm. In order to judge whether or not premature convergence is present, Lv and Hou [15] have investigated, applying the rule, that the variance of the population's fitness σ^2 is necessarily less than a threshold value *c*. The definition is described as follows:

$$\sigma^2 = \sum_{i=1}^{N} \left(\frac{f_i - f_{\text{avg}}}{f} \right)^2, \tag{1}$$

$$f = \begin{cases} \max\left\{ \left| f_i - f_{\text{avg}} \right| \right\} & \text{if} \quad \max\left\{ \left| f_i - f_{\text{avg}} \right| \right\} > 1, \\ 1 & \text{else}, \end{cases} \tag{2}$$

where f_i is the fitness of the *i*-th particle; f_{avg} – the fitness average of the particle swarm; N – the particle swarm size; f – the normalizing factor.

Liu and Lin [12] have investigated defining not one standard for the reasonable value of the threshold value *c* and assuming the possibility that σ^2 approaches zero when the algorithm converges to the global optimal solution by a numerical experiment. According to further numerical experiments, the premature convergence and the variance of the population's fitness σ^2 are related to some extent. When the premature convergence occurs, σ^2 does not change much in several continuous iterations. If the difference among the variances of the population's fitness σ^2 is very small, a premature convergence is likely to occur. In such case, we need to give perturbations to the current particle swarm in order to avoid the local minimum point.

Based on the above analysis, this paper uses the Anderson chaotic mapping to generate the random number, and uses the change of the variance of the population's fitness to control the portion of particles that undergo the chaos update.

2. Chaos random number generator

The behaviour of the chaotic motion is complex and analogous to a random motion, but it also has its inherent law, displaying properties, such as pseudo-randomness, ergodicity, regularity and so on. It is between some definite phenomena and a completely random phenomenon. The common logistic chaotic mapping is

(3) $$cx_{n+1} = \mu \times cx_n \times (1 - x_n),$$

where μ is the control parameter, and

$$0 \le cx_0 \le 1; \ n = 0, 1, 2, \ldots$$

The above system is completely in a state of chaos and $0 \le cx_n \le 1$ when $\mu = 4,$ as investigated by L v, L u and C h e n [16].

A n d e r s o n [17] has investigated the recurrence formula of the random number generation raised by Anderson chaotic mapping which is shown as follows:

(4) $$y_{n+1} = \begin{cases} \frac{3}{2} y_n + \frac{1}{4}, & 0 \le y_n < \frac{1}{2}, \\ \frac{1}{2} y_n - \frac{1}{4}, & \frac{1}{2} \le y_n < 1, \end{cases} \quad n = 0, 1, 2, \ldots$$

This recurrence formula can generate sequences in an infinite period. The limiting distribution of the empirical distribution is

(5) $$F(y) = (\ln(y + 1/2) + \ln(2)) / \ln(3).$$

According to the expression

(6) $$cx_i = (\ln(y_i + 1/2) + \ln(2)) / \ln(3),$$

the sequence $\{cx_i\}$ can be considered as uniform distribution on (0, 1).

The uniformity of the two kinds of chaotic mapping is compared as follows. Firstly we arbitrarily take an initial value $cx_0 = 0.48721123$ and use (3) and (6) to iterate to arrive at 30 000 figures which are denoted by the sequence $\{c_{xi}\}$. Then the interval [0, 1) is subdivided into 10 subintervals. Finally we consider the frequency of the sum of the number of figures which appears in each subinterval. The result is shown in Table 1.

Table 1. The comparison of the uniformity for the two kinds of chaotic mapping with 30 000 digits

Subintervals	The chaotic mapping in this paper		Logistic mapping	
	The number of the data	The ratio of data (%)	The number of the data	The ratio of data (%)
[0.0, 0.1)	2998	9.99	6126	20.42
[0.1, 0.2)	3003	10.01	2725	9.08
[0.2, 0.3)	2998	9.99	2185	7.28
[0.3, 0.4)	3000	10.00	2058	6.86
[0.4, 0.5)	3001	10.00	1873	6.24
[0.5, 0.6)	2998	9.99	1959	6.53
[0.6, 0.7)	3003	10.01	1974	6.58
[0.7, 0.8)	2997	9.99	2996	8.99
[0.8, 0.9)	3000	10.00	2696	8.99
[0.9, 1.0)	3002	10.01	6157	20.52

We choose the initial point randomly and iterate Equations (3) and (6). Then it generates a sequence with length of 742. Finally we compare the uniformity. After many times of iteration and comparison, we can determine that the sequences generated by Anderson chaotic mapping show better uniformity. The comparison results of the two sequences which were chosen randomly are shown in Table 2.

Table 2. The comparison of the uniformity for the two kinds of chaotic mapping with 742 digits

Subintervals	The chaotic mapping in this paper		Logistic mapping	
	The number of the data	The ratio of data (%)	The number of the data	The ratio of data (%)
[0.0, 0.1)	73	0.0984	135	0.1819
[0.1, 0.2)	75	0.1011	63	0.0849
[0.2, 0.3)	74	0.0997	64	0.0863
[0.3, 0.4)	74	0.0997	55	0.0741
[0.4, 0.5)	75	0.1011	43	0.0580
[0.5, 0.6)	73	0.0984	39	0.0526
[0.6, 0.7)	75	0.1011	62	0.0836
[0.7, 0.8)	74	0.0997	53	0.0714
[0.8, 0.9)	74	0.0997	75	0.1011
[0.9, 1.0)	75	0.1011	153	0.2062

It can be seen from Table 2 that although the length of the sequence is smaller, the one generated by Anderson chaotic mapping maintains a higher level of uniformity.

When the optimal solution of the optimization problem is not located on the edge of the optimal space, Anderson chaotic mapping is superior to the logistic chaotic mapping. Since the distribution of the solution of the optimization problem cannot be known in advance for practical use, the initial particle swarm must be of uniform distribution as far as possible in the optimization space. Hence, the random number generated needs to have good uniformity.

3. Standard PSO algorithm

PSO algorithm adopts the concept of the particle swarm and evolution and evaluates the particle according to the fitness value. Each particle corresponds to one possible solution of the problem. It does not have any volume or mass, but has a position and velocity. The value of the objective function corresponding to the position of the particle is the fitness of the particle. The algorithm firstly generates a set of initial particles, then it iterates each particle, and finally it obtains one solution. For each iteration it realizes the particle update by tracking the personal best (pbest) and global extreme (gbest).

The velocity and the position of a standard PSO are:

$$v(t+1) = w \times v(t) + c_1 \times \text{rand1} \times (\text{pbest} - x(t)) + c_2 \times \text{rand2} \times (\text{gbest} - x(t)), \tag{7}$$

$$x(t+1) = x(t) + v(t+1). \tag{8}$$

In the above equations, $v(t)$ and $x(t)$ are the particle velocity and particle position in the t-th step, respectively, rand1 and rand2 are independent random numbers within the interval [0, 1]; c_1 and c_2 are learning factors with a value 2; w is the inertia weight with a value between 0.1 and 0.9 [15]. Shi and Eberhart [18] have determined that if w decreases linearly along with an increase in the iterations, the convergence of the algorithm would be dramatically improve. That is

$$w = w_{\max} - t \times (w_{\max} - w_{\min}) / \mathrm{MaxDT}, \tag{9}$$

where $w_{\max}$ and $w_{\min}$ are the maximum and minimum of the weight, respectively. MaxDT is the maximum number of iterations and t is the current number of iterations.

In order to reduce the possibility that the particle leaves the optimization space, the range of the velocity value and position value must be restricted. Generally, if we take $|x| \le x_{\max}$, then we can take $v_{\max} = k \cdot x_{\max}$ with $0.1 \le k \le 1.0$.

4. The modified PSO

Consider the following global optimization model:

$$\min f(x) = f(x_1, x_2, \cdots, x_n). \tag{10}$$

where n is the dimension of the variable x. This paper uses real coding.

4.1. The initial population generated by chaotic mapping

An n-dimensional vector cx_1 was randomly generated with a value of each element ranging between 0 and 1. For each element of cx_1, Anderson chaotic mapping is used with N–1 iterations to generate N–1 n-dimensional points which are respectively denoted by $cx_2, cx_3, \cdots, cx_N$. The n-dimensional vectors $cx_1, cx_2, cx_3, \cdots, cx_N$ were firstly converted in an order with respect to the optimization space by using (12). Then the fitness was calculated by using (10). Finally M points were selected to be the initial population.

The variable x, the value of which is not between 0 and 1, can be converted by (11) and (12) while assuming $x \in (a, b)$.

4.2. The modified iteration update

This paper uses σ^2 behavior between the iterations in order to determine whether premature convergence occurs. If the difference of σ^2 between iterations is less than a given value, for instance eps = 10^{-6}, it can be assumed that the current particle swarm needs to be disturbed. Firstly the number of particles is determined which need to be replaced, as s. This paper takes s as 61.8% of the total number of particles. Starting with the particle's position which is generated randomly, the positions of $s-1$ particles are generated by Anderson chaotic mapping. The

positions of *s* particles are used to replace the positions of *s* particles which have the worst fitness in the current particle swarm and the PSO iteration process continues.

4.3. The termination condition of the iterations

For the sake of simplicity, this paper takes the given maximum number of iterations as the termination condition. When the standard function is used to test the efficiency of the algorithm, the rule can also apply that the fitness value is up to the standard to be the convergence criterion.

4.4. The detailed flow of the modified PSO

Step 1. Initialize the inertia weight w_0, the learning factors c_1 and c_2, the population size N, the maximum number of iterations MaxDT, the number of dimensions D, the precision eps = 10^{-6}, the given optimization space $[x_{\min}, x_{\max}]$, and the velocity limit $v_{\max}$.

Step 2. Firstly generate a D-dimension space particle, then use (4) and (6) to get $N-1$ particles. Finally use (12) to get N D-dimension particles in the optimization space which are denoted by x_i, where $i=1,2,\cdots,N$. Analogously initialize the flight velocity letting the number of iterations equal to 0.

Step 3. Firstly x_i is substituted into the objective function to calculate the fitness f_i, then determine the global optimal position of the particle swarm (gbest) and the optimal position (pbest$_i$) that the particle goes through in itself, where $i=1,2,\cdots,N$.

Step 4. w decreases according to (9). The particle position and velocity are updated according to (7) and (8), respectively. The iterations increase by one. Update (gbest) and (pbest). Calculate the difference between the fitness variance of the current particle swarm and that of the previous iteration. If the absolute value of the difference is less than eps, go to Step 5. Otherwise go to Step 6.

Step 5. Calculate *s*. Being analogous to the operation which generates the position of the initial particles, generate *s* new particles to replace *s* particles which have the worst fitness. Go to Step 4.

Step 6. If the number of the iterations is less than MaxDT, go to Step 4. Otherwise, go to Step 7.

Step 7. Output the final results (gbest) and (pbest).

5. Numerical simulation

In order to test the performance of the modified algorithm, five nonlinear standard functions are chosen as shown in Table 3. f_1 and f_2 are unimodal high-dimensional functions. f_3 and f_4 are multimodal low-dimensional functions.

Table 3. The standard test functions

Function	Dimension	Domain	Convergence criteria	Optimal value
$f_1 = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^n$	10^{-6}	0
$f_2 = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]^n$	100	0
$f_3 = \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[–5.12, 5.12]	100	0
$f_4 = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]^n$	10^{-6}	0
$f_5 = \frac{\sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2} + 0.5$	2	$[-100, 100]^n$	0	0

The theoretical optimal solution is difficult to be obtained, so this paper proposes a new convergence criterion.

The algorithm presented in this paper is compared with the standard PSO. We set $c_1 = c_2 = 2$, and set the population size of high-dimensional functions as 40 and that of low-dimensional functions as 10. The variable dimension, domain and convergence criteria are shown in Table 1. The inertia weight *w* linearly decreases from 0.9 up to 0.2. The proportion of the particles which are generated by chaotic mapping is 61.8% and set eps = 10^{-6}. The algorithm in this paper relies on the chaos to jump off the local minimum. So the more iterations there are, the better the obtained results will be. We set the maximum iterations MaxDT = 10 000, and run randomly 100 times. In order to compare the search efficiency of the algorithm, we adopt the following criterion.

1. With a successful search, take the average value of the optimal values and denote it as mB.

2. Denote the ratio of the successful search as Ir.

The results are shown in Table 4.

Table 4. The results of function search

Functions	mB of the standard PSO/ convergence criteria/optimal value	lr of the standard PSO	mB of this paper/ convergence criteria/ optimal value	lr of the algorithm presented in this paper
f_1	/10^{-6}/ 0	0	3.5651×10^{-15}/ 10^{-6}/0	1.0
f_2	/100/ 0	0	32.555/ 100/ 0	0.97
f_3	86.4181/ 100/ 0	0.32	35.0623/ 100/ 0	1.0
f_4	/10^{-6}/ 0	0	5.6199×10^{-14}/ 10^{-6}/ 0	0.25
f_5	0/ 0/ 0	0.18	0/ 0/ 0	0.58

On the whole, the algorithm presented in this paper is better than the standard PSO. With the former, the dimensions of the first four test functions are high. The higher the complexity of the test function is, the smaller the range of the corresponding optimization space is. Although the particle population size is 40, slightly larger than dimension of 30, the algorithm presented in this paper still shows better performance. The reason is the high level of uniformity of the sequence generated by Anderson chaotic mapping and it benefits by jumping off the local minimum point.

For f_5, the ratio for which this paper's algorithm converges to zero, is 0.72. The image of the function that is near to the optimal point, is shown in Fig. 1.

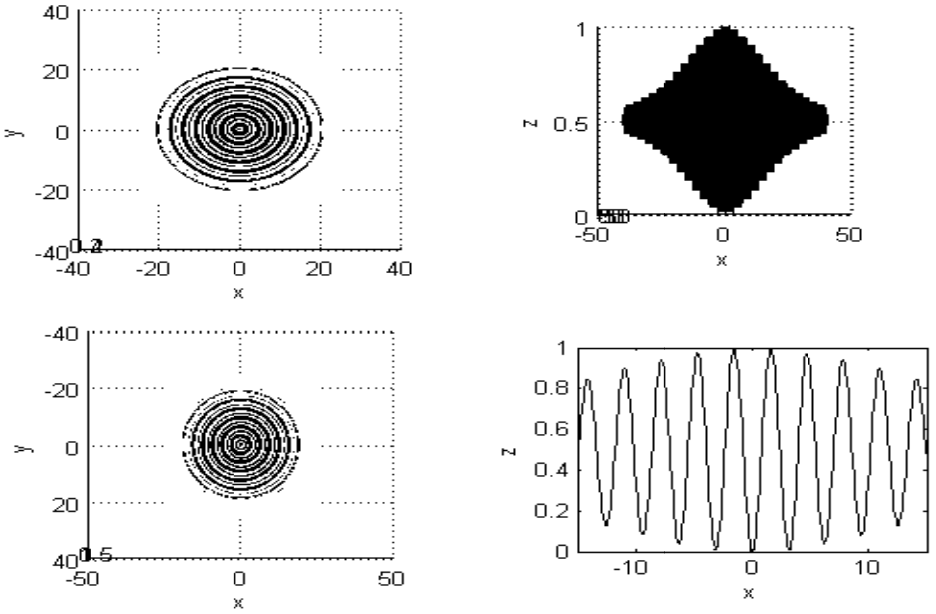

Fig. 1. The image showing that the function f_5 is near to the optimal point

The upper left figure is the plan view. The upper right figure is the side view. The left lower figure is the upward view. The right lower figure is the sectional view in *xOz* plan.

After the function f_5 is optimized 50 times, using the algorithm presented in this paper, we get the optimization result which only contains two values, 0 and 0.0097159. The optimization result is the global optimal point or the global sub-optimal point. There are two reasons for the optimization result to contain 0.0097159. The first reason is that the ratio of the area of a valley shape which covers the global optimal point to the area of the search range is less than 0.000256. The second reason is that the algorithm mainly searches in some range of current global optimal point.

In order to demonstrate the inherent superiority of the algorithm presented in this paper, three-dimensional graphs and contour maps near the optimal points are given when the dimension of the functions f_1, f_2, f_3 and f_4 equals 2. The symbol

"+" is used to denote the optimal point and the arrows are used to highlight it, as shown in Fig. 2.

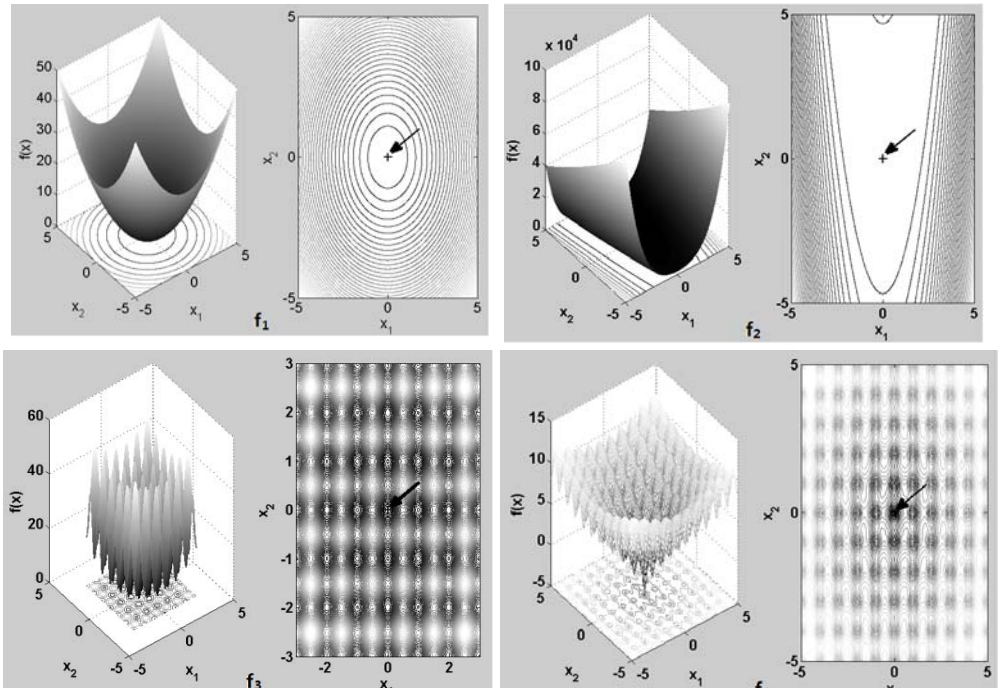

Fig. 2. The graphs of the functions f_1, f_2, f_3 and f_4 near the optimal points

In Fig. 2 the optimal point of function f_1 is the point of tangency between a sphere and a plane. The optimal point of function f_2 is located in the flat area which changes more slowly in X_2 direction than in X_1 direction. The function f_3 is a multimodal function. Although the dimension is 30 when it is optimized, the successful convergence rate of the algorithm reaches 100% because of the small optimization range. The function f_4 has 25 local optimal points in the domain [–1, 1]×[–1, 1]. Table 3 shows that the successful convergence rate of the algorithm for the function f_4 is 0.25. The reason is that the optimization domain $[-600, 600]^{30}$ is too large and the dimension of the function is too high. When the optimization domain of the function f_4 is reduced to $[-5.12, 5.12]^{30}$, the successful convergence rate of the algorithm increases to 100%.

The state of successful convergence was chosen, and the convergence rate was compared between the algorithm and the standard particle swarm optimization. The results are shown in Fig. 3.

From Fig. 3 it can be seen that the the standard PSO has faster convergence rate, but it is apt to get entrapped in premature convergence and cannot search the optimal value. The algorithm presented in this paper introduces the Anderson chaotic mapping to perform perturbation, and it can improve the optimization capability of the algorithm by increasing the iterations.

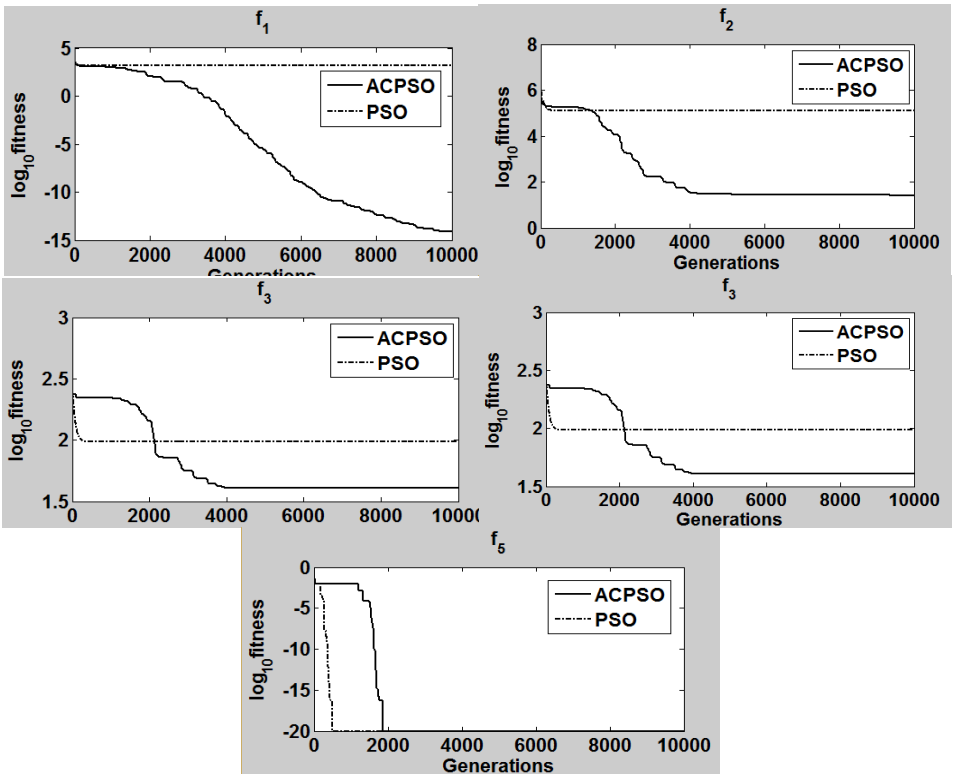

Fig. 3. Comparison of the convergence between the two algorithms for different test functions

6. Conclusion

The test results of the above standard functions show that the algorithm which initializes the particles by Anderson chaotic mapping and uses the change of fitness variance to control the update of a portion of particles has a beneficial convergence effect.

The convergence speed of PSO is not very stable because of the characteristics of the standard functions and the inherent characteristic of the PSO algorithm. Additionally, random factors exist in the initialization and update process of the particles. But the optimization capability of the algorithm presented in this paper will increase along with the increase of the iterations.

We will further analyze the effect of the parameters on the algorithm's performance, such as the size of the particle swarm, the proportion of the particles which undergo chaos mapping update while performing perturbations and iterations.

***Acknowledgements*:** The authors would like to thank the people in the Institute of Petroleum Engineering Technology, Zhongyuan Bureau of Petroleum Exploration for their great help. This paper is supported by the National Natural Science Foundation of China (61273179 and 61572084) and the Natural Foundation of Hubei Province of China (2013CFA053). It is also supported by the Educational Commission of Hubei Province of China (Q20121216) and the scientific research projects of Hubei Province of China (B2015449).

References

1. Kennedy, J., R. C. Eberhart. Particle Swarm Optimization. – In: Proc. of IEEE International Conference on Neutral Networks, Perth, Australia, [s. n.], 1995, pp. 1942-1948.
2. Clerc, M., J. Kennedy. The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. – IEEE Transactions on Evolutionary Computation, Vol. **6**, 2002, No 1, pp. 58-73.
3. Chatterjee, A., P. Siarry. Computational Intelligence in Image Processing. Berlin, Springer, 2013, pp. 57-65.
4. Vaisakh, K., P. Praveena, S. R. M. Rao, K. Meah. Solving Dynamic Economic Dispatch Problem with Security Constraints Using Bacterial Foraging PSO-DE Algorithm. –Electrical Power and Energy Systems, Vol. **5**, 2012, No 3, pp. 56-67.
5. Babu, M. S. P., M. Ramjee, S. S. V. N. L. Narayana, S. N. V. R. Murty. Sheep and Goat Expert System Using Artificial Bee Colony (ABC) Algorithm and Particle Swarm Optimization (PSO) Algorithm. – In: Proc. of IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS), Beijing, 2011, pp. 51-54.
6. Niknam, T. An Efficient Hybrid Evolutionary Algorithm Based on PSO and HBMO Algorithms for Multi-Objective Distribution Feeder Reconfiguration. – Energy Conversion and Management, Vol. **50**, 2009, pp. 2074-2082.
7. Lin, T., P. Wu, F. Gao et al. Study on SVM Temperature Compensation of Liquid Ammonia Volumetric Flowmeter Based on Variable Weight PSO. – International Journal of Heat and Technology, Vol. **33**, 2015, No 2, pp. 151-156.
8. Piancastelli, L., L. Frizziero, G. Zanuccoli et al. Design and Heuristic Optimization of Low Temperature Differential Stirling Engine for Water Pumping. – International Journal of Heat and Technology, Vol. **31**, 2013, No 1, pp. 9-16.
9. Wang, C. A Rapid Auto-Focus Method in the Telephoto Lens. – Review of Computer Engineer Studies, Vol. **2**, 2015, No 2, pp. 13-18.
10. Wang, T., H. Yan, S. Zhong et al. Research of Fire Alarm System Based on Extension Neural Network. – Review of Computer Engineer Studies, Vol. **2**, 2015, No 1, pp. 9-14.
11. Kundra, H., M. Sood. Cross-Country Path Finding Using Hybrid Approach of PSO and BBO. – International Journal of Computer Applications, Vol. **7**, 2010, pp. 15-19.
12. Liu, H., Y. Lin. A Hybrid Particle Swarm Optimization Based on Chaos Strategy to Handle Local Convergence. – Computer Engineering and Applications, Vol. **42**, 2006, No 13, pp. 77-79.
13. Yang, X.-S. Chaos-Enhanced Firefly Algorithm with Automatic Parameter Tuning. – International Journal of Swarm Intelligence Research, Vol. **2**, 2011, pp. 1-11.
14. Lu, Y., X. Liu. A New Population Migration Algorithm Based on the Chaos Theory. – In: Proc. of IEEE 2nd International Symposium on Intelligence Information Processing and Trusted Computing (IPTC), Bangkok, Thailand, 2011, pp. 147-158.
15. Lv, Z., Z. Hou. Particle Swarm Optimization with Adaptive Mutatio. – ACTA ELECTRONICA SINICA, Vol. **32**, 2006, No 3, pp. 416-420.
16. Lv, J., J. Lu, S. Chen. Chaotic Time Series Analysis and its Application. Wuhan, Wuhan University Press, 2002, pp. 34-35.
17. Anderson, R. Industrial Cryptography. IEEE REV, 1996, pp. 118-120.
18. Shi, Y., R. C. Eberhart. A Modified Swarm Optimizer. – In: Proc of IEEE International Conference of Evolutionary Computation. Anchorage, Alaska: IEEE Press, 1998, pp. 73-81.