An Improved θ-PSO Algorithm with Mutation

Weimin Zhong[1] Jianliang Xing[1,2] Hongwei Ge[1] Feng Qian[1]

[1]State-Key Laboratory of Chemical Engineering, Automation Institute of East China University of Science and Technology, Shanghai 200237 P.R. China
[2]Sinopec Yangzi Petrochemical Company LTD, Nanjing 210048 P.R. China

Abstract

In our previous work, a new PSO algorithm called θ-PSO based on phase angle was put forward, which has better optimization performance than standard PSO algorithm when dealing with some simple benchmark functions. But this algorithm may easily stick in the local minima when handling some complex or multi-mode functions. In this paper, an improved θ-PSO with mutation operator is studied. And this improved algorithm has better optimization performance when solving some complex benchmark functions. Benchmark testing shows that this improved θ-PSO algorithm can overcome the local minima and achieve the goal of global minimum in limited iterations.

Keywords: Particle swarm optimization (PSO), Phase angle, Benchmark function, Mmutation

1. Introduction

PSO is one of the famous evolutionary computation techniques introduced by Kennedy and Eberhart in 1995[1]-[2]. It is a population-based search algorithm which is initialized with a swarm of random particles. PSO makes use of a velocity vector to update the current position of each particle in the swarm under the rules: (1) maintaining own inertia; (2) using own personal best solution and (3) based the global best solution. The velocity vector is updated based on the history information gained by the swarm. And the positions of the swarm are updated to search for better positions according to the updated velocity vector [3]. In our early work, a new PSO algorithm, called θ-PSO was put forward [4]. In θ-PSO, increment of phase angle vector $\Delta\vec{\theta}$ replaces velocity vector $\vec{v}$ and the positions are adjusted by the mapping function of phase angles. Benchmark testing of nonlinear functions shows that θ-PSO appears to be a promising approach of function optimization. But this algorithm may easily stick in the local minima when handling some complex or multi-mode functions such as Ackly and Rastrigrin etc. In this paper, an improved θ-PSO algorithm with mutation operator is studied. And this improved algorithm has better optimization performance when solving some complex functions. Experiments results show that this improved θ-PSO can overcome the local minima and achieve the goal of global minimum in limited iterations.

2. Standard θ-PSO algorithm

In θ-PSO, the increment of phase angle replaces velocity and the position is decided by the mapping of phase angle. The standard θ-PSO can be described in vector notation as follow:

$$\begin{aligned}\Delta\vec{\theta}_i(t+1) = {} & \omega * \Delta\vec{\theta}_i(t) + c_1 * \vec{r}_1(t) * (\vec{\theta}_{ib}(t) - \vec{\theta}_i(t)) \\ & + c_2 * \vec{r}_2(t) * (\vec{\theta}_g(t) - \vec{\theta}_i(t))\end{aligned} \tag{1}$$

$$\vec{\theta}_i(t+1) = \vec{\theta}_i(t) + \Delta\vec{\theta}_i(t+1) \tag{2}$$

$$\vec{x}_i(t) = f(\vec{\theta}_i(t)) \tag{3}$$

$$F_i(t) = fitnessvalue(\vec{x}_i(t)) \tag{4}$$

with $\theta_{ij} \in (\theta_{\min}, \theta_{\max})$ $\Delta\theta_{ij} \in (\Delta\theta_{\min}, \Delta\theta_{\max})$, $x_{ij} \in (x_{\min}, x_{\max})$ and f is a monotonic mapping function, $i = 1, \cdots, s$, $j = 1, \cdots, n$.

We assume the global optimal particle is not on the boundary and

s is the size of the swarm.

n is the dimension of the problem

c_1 and c_2 are acceleration coefficients

ω is the inertia weight

$\vec{r}_1(t)$ and $\vec{r}_2(t) \sim U(0,1)$

$\vec{x}_i(t)$ is the position of particle i at time t, which is decided by the mapping function f

$\vec{\theta}_i(t)$ is the phase angle of particle i at time t

$\Delta\vec{\theta}_i(t)$ is the increment of phase angle of particle i at time t

$\vec{\theta}_{ib}(t)$ is the phase angle of personal best solution of particle i at time t

$\vec{\theta}_g(t)$ is the phase angle of global best at time t

$F_i(t)$ is the fitness value of particle i at time t, which is decided by function $fitnessvalue$

$F_{ib}(t)$ is the personal best fitness value of particle i at time t

$F_g(t)$ is the global best fitness value at time t

And in this paper, we set

$\theta_{ij} \in (-\frac{\pi}{2}, \frac{\pi}{2})$, $\Delta\theta_{ij} \in (-\frac{\pi}{2}, \frac{\pi}{2})$ and

$$f(\theta_{ij}) = \frac{x_{\max} - x_{\min}}{2} \sin(\theta_{ij}) + \frac{x_{\max} + x_{\min}}{2} \quad (5)$$

3. Improved θ-PSO algorithm

Compared to basic PSO algorithm, θ-PSO algorithm has better optimization performance when dealing with some simple benchmark functions [1]. But it's difficult for basic θ-PSO algorithm to overcome the local minima when handling some complex or multi-mode functions. So in this paper, we adopt the mutation operator of genetic algorithm. If the personal fitness value has not improved compared the last iteration's result, i.e., if $F_i(t) > F_i(t-1)$, a mutation operator is introduced in the basic θ-PSO algorithm with a small probability. And the detail is as follow: if $F_i(t) > F_i(t-1)$, create a random number $d_{ij} \in (0,1)$, if $d_{ij} < Pm$, do

$$\theta_{ij}(t) = -\theta_{ij}(t) + c_3 * (\vec{r}_3(t) - 0.5) \quad (6)$$

where $Pm \in [0,1]$, c_3 is a non-negative real number, $\vec{r}_3(t) \sim U(0,1)$, and limit θ_{ij} to $(-\frac{\pi}{2}, \frac{\pi}{2})$. In this paper, a minus value of θ_{ij} is adopted because the range of phase angle is symmetrical. Simulation results show that the added disturbance item $c_3 * (\vec{r}_3(t) - 0.5)$ is effective sometime.

The improved θ-PSO algorithm can be summarized as follow:

1) Create and initialize a n-dimensional swarm (phase angle $\vec{\theta}_i(1)$) and $\Delta\vec{\theta}_i(1)$;

2) $t = 1$, calculate $\vec{x}_i(1)$ using Eq.(3), calculate the fitness value $F_i(1)$ using Eq.(4) and set $F_{ib}(1) = F_i(1)$, $\vec{\theta}_{ib}(1) = \vec{\theta}_i(1)$, $F_g(1) = \min F_i(1)$, and set $\vec{\theta}_g(1)$ equal to the panes angles corresponding to $\min F_i(1)$; Then set $t = 2$;

3) Update $\Delta\vec{\theta}_i(t)$ using Eq. (1), and limit $\Delta\vec{\theta}_i(t)$ to $(\Delta\theta_{\min}, \Delta\theta_{\max})$;

4) Update $\vec{\theta}_i(t)$ using Eq. (2), and limit $\vec{\theta}_i(t)$ to $(\theta_{\min}, \theta_{\max})$;

5) Update $\vec{x}_i(t)$ using Eq. (3);

6) Calculate $F_i(t)$ using Eq. (4);

7) If $F_i(t) < F_{ib}(t)$, then set $F_{ib}(t) = F_i(t)$ and $\vec{\theta}_{ib}(t) = \vec{\theta}_i(t)$; If $F_i(t) < F_g(t)$, then set $F_g(t) = F_i(t)$ and $\vec{\theta}_g(t) = \vec{\theta}_i(t)$;

8) If $F_i(t) > F_i(t-1)$, create a random number $d_{ij} \in (0,1)$ $j = 1, \cdots n$. And if $d_{ij} < Pm$, do mutation operation using Eq. (6), and calculate $F_i(t)$;

9) If $F_i(t) < F_{ib}(t)$, then set $F_{ib}(t) = F_i(t)$ and $\vec{\theta}_{ib}(t) = \vec{\theta}_i(t)$; If $F_i(t) < F_g(t)$, then set $F_g(t) = F_i(t)$ and $\vec{\theta}_g(t) = \vec{\theta}_i(t)$;

10) Set $t = t + 1$, go back to step 3 if stopping condition is not true.

4. Benchmark functions test

Standard θ-PSO has better performance than standard PSO when dealing with function Camel, Levy F3, Sphere and Jason. Generally speaking, Standard θ-PSO can obtain the global optimal in hundreds iterations when handling some simple functions. But this algorithm may still easily stick in the local minima sometime when handling complex multi-mode functions such as Rosenbrock, Schwefel, Rastrigrin and Ackly. In this paper, we use these four benchmark functions to demonstrate our improved algorithm. The basic information of the functions is listed in table 1.

Function Rosenbrock is a classical complex optimization case whose global optimal is located at a flat, long and narrow valley. This function gives little information and it's hard to identify the search direction. Function Rastrigrin is a very difficult case that has thousands of local minima. There is few algorithms can obtain the global minimum 0 at $(-420.9687, \cdots -420.9687)$ of function Schwefel. Function Ackly is a multi-mode case with a lot of cloughs, and the local minima are located everywhere. All these four functions are complex, full of local minima.

First, basic θ-PSO algorithm is tested by these four functions. According to the results of [4], in the following tests of this paper, we set $w = 0.6$, $c_1 = 1.7$ and $c_2 = 1.7$, which can obtain better optimization performance. And in this case, the maximum iteration number is fixed to 10000 and the swarm size is 40. Each optimization experiment is run 20 times with random initial value of θ and $\Delta\theta$. Our testing computer is IBM T60 notebook PC with 2 CPU at 1.66GHz and 512M memory, the operating system is Window XP, and the program is coded by Matlab 6.5. The test results are listed in table 2. And we can see that basic θ-PSO algorithm can not get the optimal in the fixed iterations.

Function	Formula	Dim	Range	Optimal
Rosenbrock	$f(x) = \sum_{i=1}^{n-1} (100\ (x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	30	(-2.048, 2.048)	0
Rastrigrin	$f(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	30	(-5.12, 5.12)	0
Schwefel	$f_6(x) = 418.98n - \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$	10	(-500, 500)	0
Ackly	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	100	(-30, 30)	0

Table 1: The list of test functions.

Function	Minimum value of fitness value	Maximum value of fitness value	Average value of fitness value
Rosenbrock	11.8792	19.7983	16.5436
Rastrigrin	87.4972	151.2327	134.8754
Schwefel	1189.6	1437.9	1263.9
Ackly	17.8285	19.7033	18.8354

Table 2: Test results of standard θ-PSO.

In the following experiment, we use these four benchmark functions to test our improved θ-PSO algorithm. The stopping conditions are (1) the goal of optimization is 0.01 and (2) the maximum iteration number is 10000. The swarm size is 40. Each optimization case is run also 20 times. We will observe the effect of parameters Pm and c_3. The results are shown in table 3. From the testing results, the improved θ-PSO algorithm can achieve the goal of optimization in the limited iterations and the average optimization time is only about several seconds. And the success rate is high when Pm and c_3 are set properly. In the experiment, we find the performance is better while $Pm \le 0.01$. For function Rosenbrock, when $Pm = 0.001$ and $c_3 = 2$, the indexes are best and the average iteration number is less than 1000. For function Rastrigrin, $Pm = 0.01$ and $c_3 = 0$ are proper. For function Schwefel, it works well when $Pm = 0.005$, $c_3 = 4$ and $Pm = 0.01$, $c_3 = 4$, the average iteration numbers are about 400. For function Ackly, the convergence performance is worse than others. And from table 3, when $Pm = 0.005$ and $c_3 = 0$, all the indexes are good.

In order to illustrate the convergence of our improved θ-PSO algorithm, another experiment is done with 40 particles, $Pm = 0.005$ and $c_3 = 0$. All these 20 tests execute 10000 iterations. And the results are listed in table 4. Only one try sticks in the local minimum at the value 19.452, and other 19 cases have good convergence results with the precision at e^{-14}. Figure 1 shows the average fitness values curve of the 19 good cases. Look back at figure 1, we can see the function Ackly has lots of local minima in the area when the fitness value is between 19 and 20. If the particles can jump out this area, the improved θ-PSO algorithm can converge very quickly.

In the following experiment, we use these four benchmark functions to test our improved θ-PSO algorithm. The stopping conditions are (1) the goal of optimization is 0.01 and (2) the maximum iteration number is 10000. The swarm size is 40. Each optimization case is run also 20 times. We will observe the effect of parameters Pm and c_3. The results are shown in table 3. From the testing results, the improved θ-PSO algorithm can achieve the goal of optimization in the limited iterations and the average optimization time is only about several seconds. And the success rate is high when Pm and c_3 are set properly. In the experiment, we find the performance is better while $Pm \le 0.01$. For function Rosenbrock, when $Pm = 0.001$ and $c_3 = 2$, the indexes are best and the average iteration number is less than 1000. For function Rastrigrin, $Pm = 0.01$ and $c_3 = 0$ are proper. For function Schwefel, it works well

Function	Pm	c_3	Number of iteration to achieve the goal			Average optimization time s	Success rate
			minimum	maximum	average[1]		
Rosenbrock	0.001	1	1881	6312	3143	4.973	1
	0.001	2	862	2142	1275	1.929	1
	0.005	1	1283	2673	1785	2.824	1
	0.005	2	1090	1556	1296	2.050	1
	0.01	1	2408	6149	3897	6.166	1
	0.01	2	1789	3687	2340	3.702	1
Rastrigrin	0.001	0	1287	4908	2665	7.028	1
	0.001	1	1937	4209	2986	8.506	1
	0.005	0	2435	4859	3012	8.580	1
	0.005	1	4129	7791	5876	16.739	1
	0.01	0	1076	4734	2575	4.538	1
	0.01	1	-	-	-	-	-
Schwefel	0.001	3	4583	9043	6087	8.822	0.2
	0.001	4	619	2958	1643	2.381	1
	0.005	3	678	7647	5478	7.939	1
	0.005	4	211	1085	402	0.582	1
	0.01	3	335	7458	3690	5.347	1
	0.01	4	136	901	431	0.606	1
Ackly	0.001	0	1401	4272	2803	11.492	0.9
	0.001	1	1349	7528	2876	12.233	0.9
	0.005	0	1294	3745	1907	7.693	1
	0.005	1	1351	8071	3536	14.497	0.85
	0.01	0	1215	3850	2987	12.246	0.85
	0.01	1	1582	4700	2622	10.750	1

Table 3: Test results of improved θ-PSO.

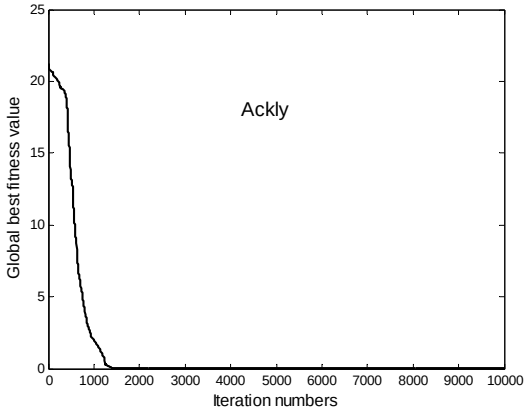

Fig.1: The average fitness values curve of function Ackly.

when $Pm = 0.005$, $c_3 = 4$ and $Pm = 0.01$, $c_3 = 4$, the average iteration numbers are about 400. For function Ackly, the convergence performance is worse than others. And from table 3, when $Pm = 0.005$ and $c_3 = 0$, all the indexes are good.

In order to illustrate the convergence of our improved θ-PSO algorithm, another experiment is done with 40 particles, $Pm = 0.005$ and $c_3 = 0$. All these 20 tests execute 10000 iterations. And the results are listed in table 4. Only one try stick in the local minimum at the value 19.452, and other 19 cases' global best fitness values have good convergence results with the precision at e^{-14}. Figure 5 shows the average fitness values curve of the 19 good cases. Look back at figure 4, we can see the function Ackly has lots of local minima in the area when the fitness value is between 19 and 20. If the particles can jump out this area, the improved θ-PSO algorithm can converge very quickly.

The effect of swarm size is tested and the results are shown in table 5. As the samethe stopping conditions are (1) the goal of optimization is 0.01 and (2) the maximum iteration number is 10000. The swarm size 20, 40, 60, 80 and 100 are used to demonstrate the optimization performance. Generally speaking, with the increasing swarm size, the minimum, maximum and average numbers of iteration to achieve the goal decrease. And the corresponding average optimization time increase. But the swarm size has little affection on the success

[1]The average number of iteration to achieve the goal and the average optimization time are the average value of successful cases.

Convergence value	19.452	$6.8390\,e^{-14}$	$6.4837\,e^{-14}$	$5.7332\,e^{-14}$	$5.0626\,e^{-14}$	$4.3251\,e^{-14}$	$3.9968\,e^{-14}$
Times	1	1	3	5	6	3	1

Table 4: The results of Ackly.

Funciton	Swarm size	*Pm*	c_3	Number of iteration to achieve the goal			Average optimization time s	Success rate
				minimum	maximum	average		
Rosenbrock	20	0.001	2	1065	3908	1998	1.559	1
	40	0.001	2	862	2142	1275	1.929	1
	60	0.001	2	553	1633	999	2.259	1
	80	0.001	2	508	1416	818	2.454	1
	100	0.001	2	501	1080	790	2.932	1
Rastrigrin	20	0.01	0	1256	8571	3477	3.139	0.95
	40	0.01	0	1076	4734	2575	4.538	1
	60	0.01	0	1037	3117	1809	4.693	1
	80	0.01	0	904	5060	2117	7.365	1
	100	0.01	0	829	5734	2125	9.194	1
Schwefel	20	0.01	4	308	1407	747	0.538	1
	40	0.01	4	136	901	431	0.606	1
	60	0.01	4	91	692	321	0.668	1
	80	0.01	4	97	838	322	0.885	1
	100	0.01	4	105	484	270	0.927	1
Ackly	20	0.005	0	1894	5569	3034	6.662	0.75
	40	0.005	0	1294	3745	1907	7.693	1
	60	0.005	0	1079	4792	2183	14.019	0.9
	80	0.005	0	1043	6835	2013	16.446	1
	100	0.005	0	957	4261	2006	20.235	1

Table 5: The list of test results of improved θ-PSO.

rate. Function Ackly is something special, compared to other three functions, the swarm size effects the maximum and average numbers of iteration to achieve the goal with some randomicity for the thousands of local minima.

5. Conclusions

An improved θ-PSO algorithm with mutation operator is put forward, which will jump out the local minima readily by adjusting the parameters *Pm* and c_3 properly. Simulation results of four complex or multi-mode benchmark functions show that, this improved algorithm can obtain the goal of global optimal in limited iterations with high success rate. But this algorithm still needs further study. For example, in our experiments, to function Ackly, c_3 should be set to near zero, while to function Schwefel, when c_3 is about 4, the optimization performance is good. So how to set the parameter c_3 according to the testing function is an interesting research direction.

Acknowledgement

This work is partially supported by National Science Fund for Distinguished Young Scholars (Grant No. 60625302), Major State Basic Research Development Program of China (973 Program, Grant No. 2002CB312200), National High-Tech Research and Development Program of China (863 Program, Grant No. 20060104Z1081) and Major State Basic Research Development Program of Shanghai (Grant No. 05DJ14002).

References

[1] J. Kennedy, R.C. Eberhart, *Swarm Intelligence*, Morgan Kaufmann Publishers, Inc. San Francisco, CA, 2001.

[2] R.C. Eberhart, J. Kennedy. A new optimizer using particle swarm theory. *Prof. of the 6th international symposium on Micro Machine and*

Human Science, Nagoya, Japan, IEEE Service Center, Piscataway, NJ, pp.39-43, 1995.

[3] S.S. Fan, E. Zahara**,** A hybrid simplex search and particle swarm optimization for unconstrained optimization. *European Journal of Operational Research*, 181(2):527-548, 2007.

[4] W.M. Zhong, S.J. Li, F. Qian, θ-PSO: A new strategy of particle swarm optimization. *Journal of Zhejiang University SCIENCE*, submitted.