International Journal of Control and Automation
Vol.7, No.5 (2014), pp.353-364
http://dx.doi.org/10.14257/ijca.2014.7.5.35

Dynamic Adjustment Strategies of Inertia Weight in Particle Swarm Optimization Algorithm

Yang Xianfeng[1] and Liu ShengLi[2]

[1]School of Information Engineering, Henan Institute of Science and Technology, Henan Xinxiang 453003, China
[2]HEBI College of Vocation and Technology, Henan Hebi 458030, China

[1]ttl10@126.com, [2]82594095@qq.com

Abstract

The high search speed and efficiency, and simple algorithm of particle swarm optimization algorithm make it suitable for actual-value processing. Starting from the angle of weight, this paper studies several improved particle swarm optimization algorithms and divides the improvement into three types as linear decreasing weight strategy, self-adaptive weight strategy and random weight strategy. Furthermore, this paper also demonstrates the principles of these three improved algorithms and tests and analyzes the three algorithms with test function. It is suggested by the result of tests that random weight strategy can make the algorithm more stable, linear decreasing weight strategy can improve the effect of optimization, while self-adaptive weight strategy can accelerate the convergence. However, the operation of self-adaptive weight strategy takes obviously more time than that of the other two strategies.

Keywords: *particle swarm, inertia weight, optimum solution*

1. Introduction

Ever since it was put forward, particle swarm optimization (PSO) has gone through many transformations and improvements, and analyses and experiments conducted by all kinds of researchers, including mathematicians, engineers, physicists, biologists and psychologists. Lots of research achievements and experience have provided many reasonable assumptions and reliable foundations for the development of particle swarm optimization and pointed out a new direction for the practical industrial application. Over the past decade, research on PSO has stretched out and extended beyond optimization. In terms of the direction of research, PSO algorithm can be divided into four parts, namely, algorithm mechanism analysis, algorithm performance improvement, algorithm application and discrete PSO algorithm [1]. Algorithm mechanism analysis mainly focuses on the convergence, complexity and parameter setting of PSO algorithm. Algorithm performance improvement mainly improves defects and shortcomings of the original PSO algorithm, with the aim to improve the performance of the original PSO algorithm or standard PSO algorithm in certain aspects. At present, the improvement of technology and technique mainly increases the diversity of algorithm and reinforces technologies of local search and integration with other intelligent optimization algorithm; the research on PSO algorithm application primarily focuses on how to utilize PSO algorithm to solve questions in engineering technology, economy and society that need optimization, and many of which are multi-objective questions, restricted questions, dynamic questions and a large number of practical application questions; discrete PSO algorithm is mostly about the optimization of discreteness and optimum solution of PSO algorithm. The original PSO algorithm mainly solves the optimization of continuity, while since discrete questions are special, the solution of this kind of questions depend on some special technologies of PSO algorithm [2]. Two of the primary questions to be studied in this situation are discrete binary system and general combinational optimization.

2. Particle Swarm Optimization Algorithm

2.1. PSO Principles

$$v_i(n+1) = v_i(n) + c_1 r_1 (p_i - x_i(n)) + c_2 r_2 (p_g - x_i(n)) \tag{1}$$

$$x_i(n+1) = x_i(n) + v_i(n) \tag{2}$$

In the equations, i=1, 2, …, m represents different particles respectively. c_1 and c_2 are learning factors (also known as acceleration coefficients) greater than zero. Normally, $c_1 = c_2$ =2; r_1 and r_2 are random numbers between [0, 1]; n is the number of iterations, namely, the step of particle flight [3, 4].

2.2 The Flow of PSO Algorithm

See the steps of basic PSO algorithm below [5, 6]:

Step 1: Initialize particle swarm, including group size, and the initial velocity and position of particles;

Step 2: calculate the fitness of every particle, save the best position Pbest and fitness of every particle, and select the position of particle with perfect fitness from the group as Gbest;

Step 3: update the velocity and position of every particle with Equation (1) and Equation (2);

Step 4: calculate the fitness of every updated particle and compare the fitness of every particle with it in the best position. If the former one is better, the present position should be deemed as Pbest of the particle;

Step 5: compare the fitness of every particle with Gbest, if the former one is better, update it as the new Gbest;

Step 6: judge if search result meets the preset end condition, which is normally perfect fitness or reaching the preset maximum iterations, if the answer is no, go back to Step 3; if the answer is yes, stop iteration and output the optimum solution.

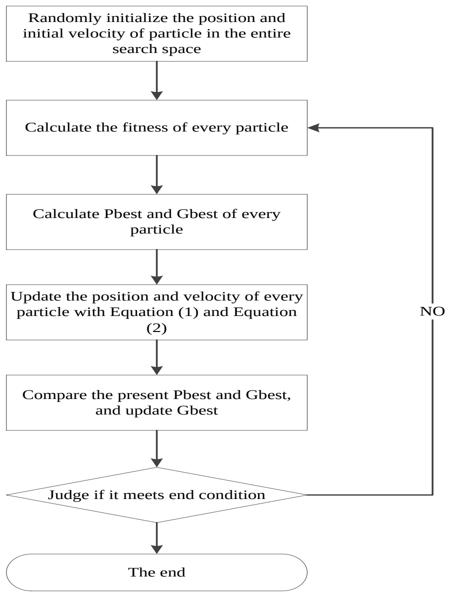

Randomly initialize the position and initial velocity of particle in the entire search space
Calculate the fitness of every particle
Calculate Pbest and Gbest of every particle
Update the position and velocity of every particle with Equation (1) and Equation (2)
Compare the present Pbest and Gbest, and update Gbest
Judge if it meets end condition
NO
The end

Figure 1. PSO algorithm flow

3. Standard PSO Algorithm

3.1. Inertia Weight

Inertia weight ω is used to control the influence of the previous velocity on the present velocity of particle, and it will affect the global and local search function of particle.

$$\omega = \omega_{\max} - n\frac{\omega_{\max} - \omega_{\min}}{n_{\max}} \tag{3}$$

$\omega_{\max}$ and $\omega_{\min}$ are respectively the maximum and minimum inertia weight. n is the number of previous iteration, $n_{\max}$ is the total number of iteration of the algorithm. ω often varies between [0.4, 0.9], and decreases with the increase of iteration [7].

3.2 Features of PSO Algorithm

To sum up, the features of PSO algorithm mainly include the following [8, 9]:

a. Concise and explicit algorithm principles, easy to realize, no need to establish complex mathematic models.

b. High velocity and precision in convergence.

c. Group search enables memory and reserves local and global optimum solution.

d. Collaborative search guides search with local individual information and global group information.

Improvements are needed in the following aspects:

a. Poor local search function, low precision.

b. Easy to be stuck in local extreme, possible failure to guarantee the global optimum solution.

c. Dependence on parameters.

4. Analysis and Selection of Algorithm Parameters

4.1. Parameter Analysis

As an important content in the study of PSO algorithm, parameter setting has huge influence on the result of algorithm optimization. See the comprehensive analysis below [10].

(1) Number of Particle Group N

N is an integer parameter. When N is small, convergence becomes fast, but will be stuck into local optimum very easily; when N is large, PSO algorithm shows perfect optimization function, but convergence becomes quite slow. Normally speaking, the number of particle group depends on specific questions and is mainly between 10 and 50.

(2) Maximum Velocity $v_{\max}$

$v_{\max}$, a vital parameter, decides the dynamics of spatial search. When $v_{\max}$ large, particle is strong in search function, but will easily skip excellent searching space; when $v_{\max}$ small, particle shows strong development function, but will easily be stuck in local optimum, which prevents it from reaching the optimum location.

(3) Learning Factor c_1 and c_2

Learning factor has the ability of summarizing itself and learning from excellent individuals in the group, which leads to particles approaching optimum pints of the group or adjacent area. c_1 and c_2 regulates the maximum step length toward optimum individual and group respectively, and decides the influence of particle’s individual experience and group experience on its orbit.

(4) End Condition of Iteration

The setting of iteration end condition needs to take multiple aspects of performance into consideration, such as the quality of algorithm optimization and search efficiency in line with specific questions. Generally speaking, when algorithm reaches the maximum iteration, the preset accuracy of calculation or acceptable solutions, iteration stops, and the optimum solution is out.

(5) Inertia Weight ω

Inertia weight decides to which extent the present particle velocity is inherited and is used to control the ability of algorithm in development and search.

The selection of parameter will affect the performance and efficiency of algorithm. Since the size of parameter space differs and there’s certain correlation among parameters, there are no general methods of determining optimum parameter in practical application.

5. Strategies of Improving PSO Algorithm Weight

5.1. Brief Introduction of Several Test Functions

(1) Griewank function formula:

$$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{4}$$

It can be obtained from the image of the function, that the function obtained the minimum value of 0 in a certain range.

(2) Rastrigin function formula:

$$f(x) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10) \tag{5}$$

Obtained from the function image, the function obtained the minimum value of 0 in a certain range.

(3) Schaffer function formula:

$$f(x,y) = 0.5 - \frac{\sin^2\sqrt{x^2+y^2} - 0.5}{[1+0.001(x^2+y^2)]^2} \tag{6}$$

Obtained from the function image, the function obtained the minimum value of -1 within a certain range.

(4) Ackley function formula:

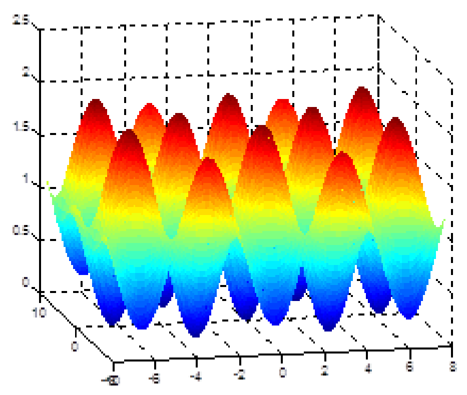
$$f(x) = -20\exp\left[-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right] - \exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right] + 20 + e \tag{7}$$

Obtained from the function image, the function obtained the minimum value of 0 in a certain range.

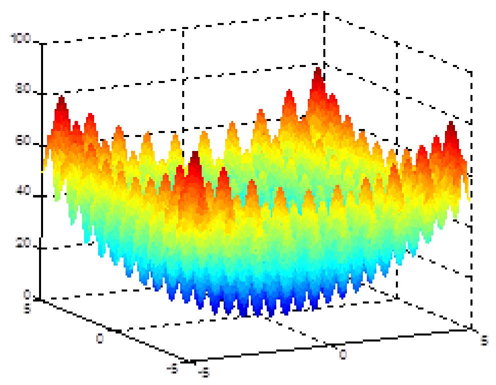
(5) Rosenbrock function formula:

$$f(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \tag{8}$$

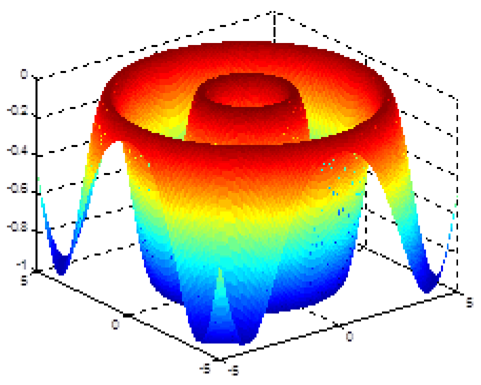
Obtained from the function image, the function obtained the minimum value of 0 in a certain range.



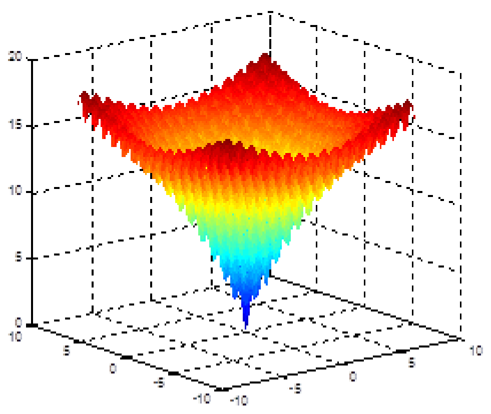
Griewank function



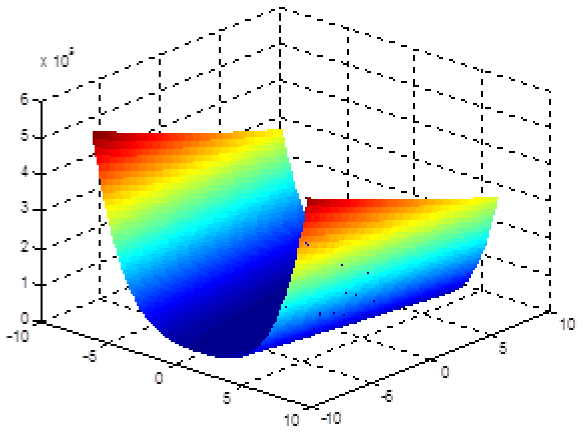
Rastrigin function



Schaffer function



Ackley function



Rosenbrock function

Figure 2. Introduction of several test functions

5.2 Three Weight Improvement Strategies

Different PSO algorithms can be achieved from different weight changing equations. Common algorithms include linear decreasing weight, self-adaptive weight and random weight, which will be elaborated separately below.

5.2.1 Linear Decreasing Weight

(1) Principle

In consideration of the fact that PSO algorithm will easily become pre-matured and generate turbulence near the global optimum solution at the later stage, linearly changing weight can be adopted to decrease inertia weight from the maximum value $\omega_{\max}$ to the minimum value $\omega_{\min}$. The changing equation of ω iteration with the algorithm is shown below.

$$\omega = \omega_{\max} - t\frac{\omega_{\max} - \omega_{\min}}{t_{\max}} \tag{9}$$

In this equation, $\omega_{\max}$ and $\omega_{\min}$ represent the maximum value and the minimum value of ω respectively; t represents the present iteration, $t_{\max}$ represents the maximum iteration. Normally, $\omega_{\max}$ =0.9, $\omega_{\min}$ =0.4.

5.2.2 Self-adaptive Weight

(1) Principle

In order to balance the global search function and local improvement function of PSO algorithm, non-linear dynamic inertia weight coefficient equation can be adopted here. See the equation below:

$$\omega = \begin{cases} \omega_{\min} - \dfrac{(\omega_{\max} - \omega_{\min})*(f - f_{\min})}{f_{avg} - f_{\min}}, f \le f_{avg} \\ \omega_{\max}, f > f_{avg} \end{cases} \tag{10}$$

In this equation, $\omega_{\max}$ and $\omega_{\min}$ represent the maximum value and minimum value of ω, f represents the present objective function value of particles, f_{avg} and $f_{\min}$ represent the average objective value and the minimum objective value of all the present particles.

5.2.3 Random Weight

(1) Principle

See ω equation below:

$$\begin{cases}\omega = \mu + \sigma * N(0,1) \\ \mu = \mu_{\min} + (\mu_{\max} - \mu_{\min}) * rand(0,1)\end{cases} \tag{11}$$

$N(0,1)$ represents random number of standard normal distribution, $rand(0,1)$ presents random number between 0 and 1.

Change curve of ω is as Figure3.

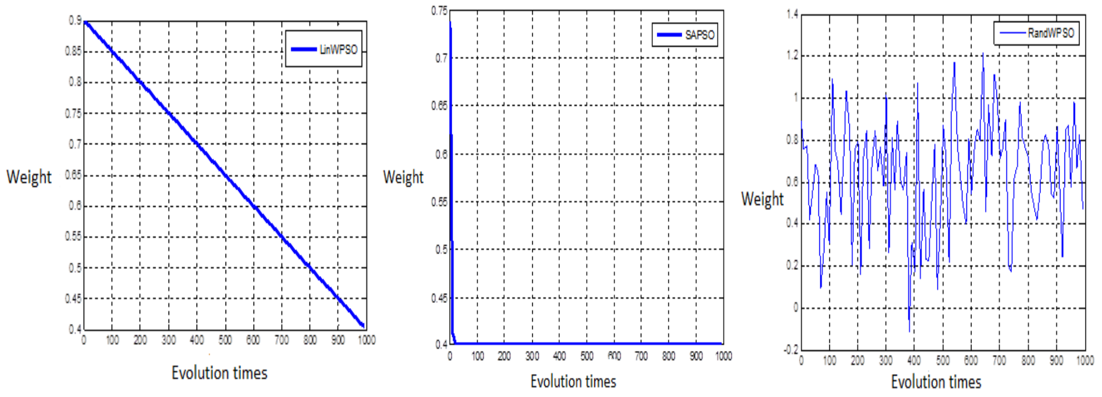


Figure 3. Three weight improvement strategies

6. Test Three Weight Improvement Strategies

6.1 Evolution Curve of Three Strategies Tests of Griewank Function

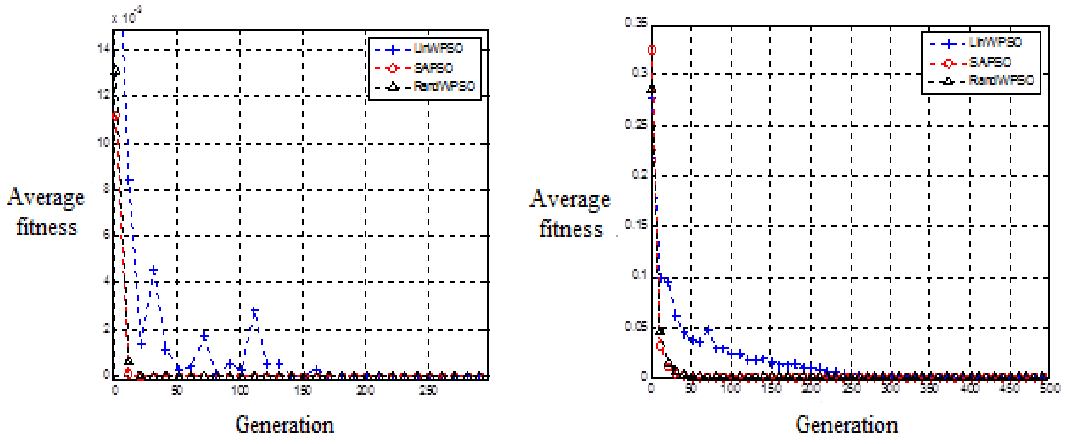


Figure 4. Griewank function dimensionality D=2 and D=10 test result

6.2 Evolution Curve of Three Strategies Tests of Rastrigin Function

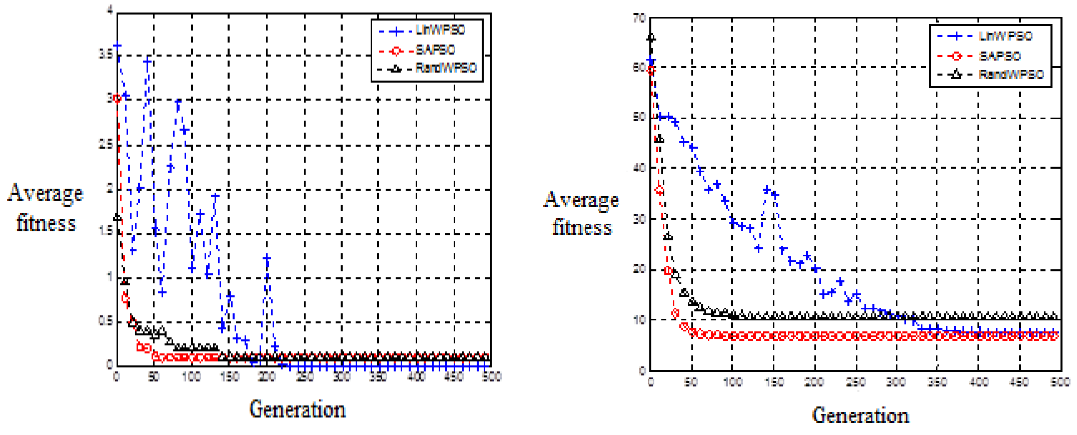


Figure 5. Rastrigin function dimensionality D=2 and D=10 test result

6.3 Evolution Curve of Three Strategies Tests of Schaffer Function

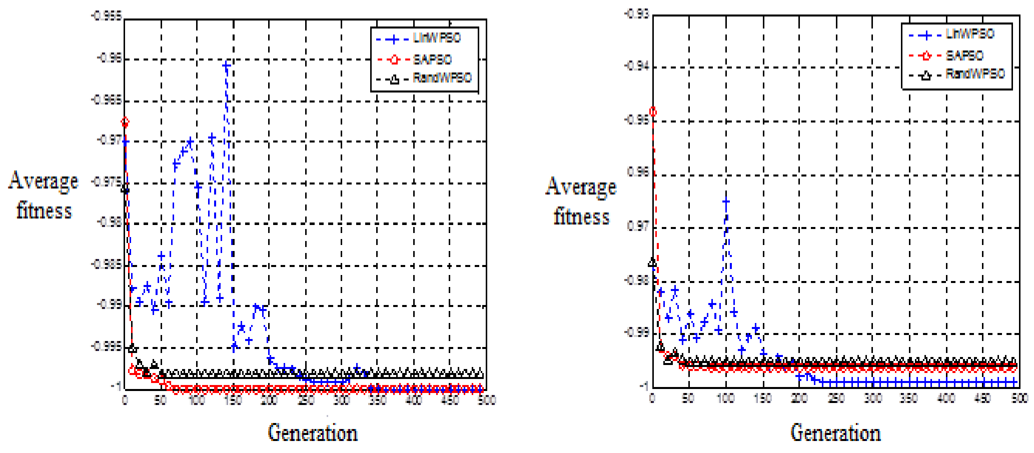
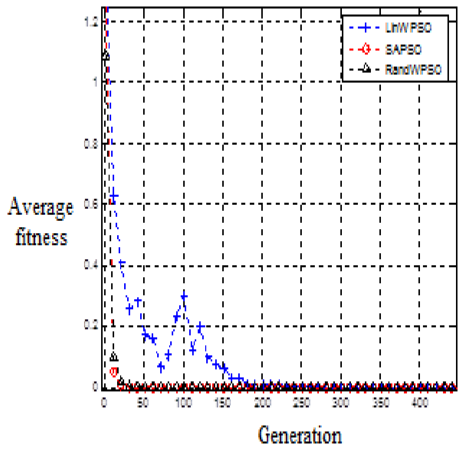


Figure 6. Schaffer function dimensionality D=2 and D=10 test result

6.4 Evolution Curve of Three Strategies Tests of Ackley Function



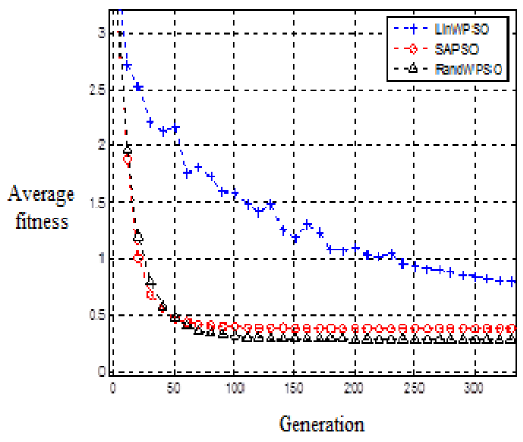
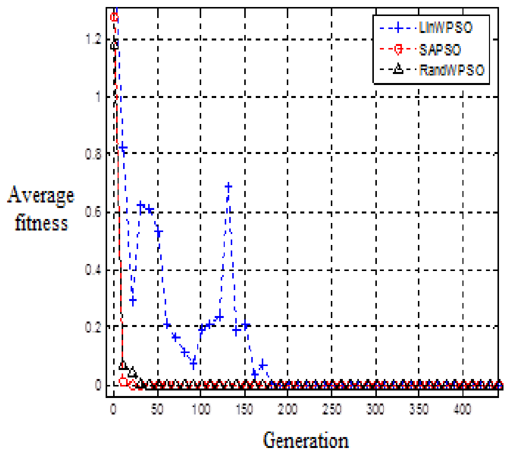


Figure 7. Ackley function dimensionality D=2 and D=10 test result

6.5 Evolution Curve of Three Strategies Tests of Rosenbrock Function



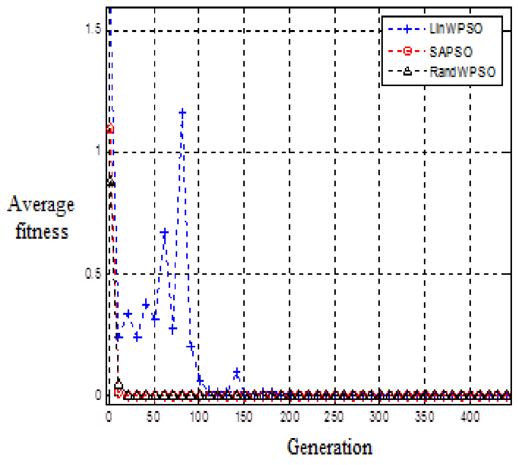


Figure 8. Rosenbrock function dimensionality D=2 and D=10 test result

6.6 Test Results and Conclusion Analysis of Three Weight Improvement Strategies

6.6.1 Test Result Data Sheet

Dimensionality D=2, test result:

Improvement Strategy	Function	Average Optimum Fitness	Average Running Time (unit: s)	Standard Deviation (6 decimals)
Griewank	LinWPSO	0	1.196	0
	SAPSO	0	1.557	0
	RandWPSO	0	1.193	0

Rastrigin	LinWPSO	0	1.075	0
	SAPSO	0.139	1.389	0.314633
	RandWPSO	0.147	1.062	0.419511
Schaffer	LinWPSO	-1	0.848	0.003072
	SAPSO	-1	1.120	0.005017
	RandWPSO	-0.995	0.857	0
Ackley	LinWPSO	0	1.276	0
	SAPSO	0	1.682	0
	RandWPSO	0	1.267	0
Rosenbrock	LinWPSO	0	0.736	0
	SAPSO	0	0.956	0
	RandWPSO	0	0.737	0

Dimensionality D=10, test result:

Improvement Strategy	Function	Average Optimum Fitness (3 decimals)	Average Running Time (unit: s)	Standard Deviation (6 decimals)
Griewank	LinWPSO	0	1.464	4.756398e-008
	SAPSO	0	1.898	6.175240e-005
	RandWPSO	0	1.468	3.958067e-008
Rastrigin	LinWPSO	10.172	1.259	2.930492
	SAPSO	7.945	1.653	3.880726
	RandWPSO	8.217	1.257	4.666770
Schaffer	LinWPSO	-0.998	0.857	0
	SAPSO	-0.996	1.134	0.004693
	RandWPSO	-0.995	0.857	0.004096
Ackley	LinWPSO	0	1.423	0.557927
	SAPSO	0.411	1.942	0.626560
	RandWPSO	0.293	1.415	0.674569
Rosenbrock	LinWPSO	0	0.745	0
	SAPSO	0	0.964	0
	RandWPSO	0	0.740	0

6.6.2 Result Analysis and Conclusion

(1) Stability

Run ten times repeatedly under random initial group, no significant changes show up in the result, the achieved optimum solution is in the same order of magnitude and shows little difference. The result is relatively stable.

(2) Optimization

All of the three improvement algorithms can optimize function very well and converge toward the optimum solution area stably. Linear decreasing improvement shows huge fluctuations in the fitness at the initial stage.

(3) Rate of Convergence

It is shown by the evolution curve that self-adaptive weight improvement is faster in convergence, but there are probably exceptions in different running functions.

(4) Running Time

Self-adaptive weight improvement strategy takes obviously more time than the other two algorithms. Therefore, when the number of iterations and repeated experiments is huge, running time will show substantial differences.

7. Conclusion

Firstly, this paper introduces the principles and optimization steps of standard particle swarms, and analyzes the influence of different parameters on algorithm optimization in details. Moreover, inertia weight, a key parameter which includes linear decreasing weight improvement, self-adaptive weight improvement and random weight improvement, is introduced. The principles of these improved algorithms are introduced, and the results of tests are analyzed. It is suggested by concrete testing graphs and data that different improvement methods will lead to different effects: random weight strategy can enhance the stability, linear decreasing weight strategy can improve the optimization, and self-adaptive weight strategy can accelerate convergence. However, self-adaptive weight strategy obviously takes more time than the other two. Therefore, the improvement of self-adaptive weight strategy has overwhelming advantages in optimization effect, convergence rate and running time. While improving these algorithms, this paper also puts forward relevant questions that need to be solved, for example, the relatively weak mathematical theory base of PSO algorithm, the poorly proved advantages and disadvantages of algorithm performance mathematically; how to optimize parameters to avoid pre-matured function and rapid convergence; the improvement of algorithm principles and the promotion of application fields, *etc.*

References

[1] T. Xiang, X. F. Liao and K. W. Wang, “An improved particle swarm optimization algorithm combined with piecewise linear chaotic map”, Applied Mathematics and Computation, vol. 190, no. 2, **(2007)**, pp. 1637-1645.

[2] J. Jie, J. C. Zeng and C. Z. Han, “Self-Organized particle swarm optimization based on feedback control of diversity”, Journal of Computer Research and Development, vol. 45, no. 3, **(2008)**, pp. 464-471.

[3] Y. X. Shen, G. Y. Wang and C. H. Zeng, “Correlative particle swarm optimization model”, Journal of Software, vol. 22, no. 4, **(2011)**, pp. 695-708.

[4] S. S. Fan and E. Zahara, “A hybrid simplex search and particle swarm optimization for unconstrained optimization”, European J of Operational Research, vol. 181, no. 2, **(2007)**, pp. 527-548.

[5] L. Liu, W. M. Zhong and F. Qian, “An improved chaos-particle swarm optimization algorithm”, J of East China University of Science and Technology: Natural Science Edition, vol. 36, no. 2, **(2010)**, pp. 267-272.

[6] Y. P. Chang, “Integration of SQP and PSO for optimal planning of harmonic filters”, Expert Systems with Applications, vol. 37, no. 3, **(2010)**, pp. 2522-2530.

[7] J. M. Xiao, J. J. Li and X. H. Wang, “Convergence analysis of particle swarm optimization and its improved algorithm based on gradient”, Control and Decision, vol. 24, no. 4, **(2009)**, pp. 560-564.

[8] X. C. Zhao, “A perturbed particle swarm algorithm for numerical optimization”, Applied Soft Computing, vol. 10, no. 1, **(2010)**, pp. 119-124.

[9] T. Xiang, X. F. Liao and K. W. Wang, “An improved particle swarm optimization algorithm combined with piecewise linear chaotic map”, Applied Mathematics and Computation, vol. 190, no. 2, **(2007)**, pp. 1637-1645.

[10] J. Jie, J. C. Zeng and C. Z. Han, “Self-Organized particle swarm optimization based on feedback control of diversity”, Journal of Computer Research and Development, vol. 45, no. 3, **(2008)**, pp. 464-471.