Improvement of Inertia Weight Declining Strategy Based on Particle Swarm Optimization Algorithm

Li-li GUO¹, Yong-LIU¹, Wei-xi WANG¹, Ya-wei LI¹, Hao-nan LI¹, Jia-qi WANG¹, Zi-wei LIU¹ and Hua-wei SUN²

¹College of Electronic Engineering, Heilongjiang University, Harbin 150080, China
²Heilongjiang Eastern Water Saving Equipment Co., Ltd., Suihua 150000, China

*Corresponding author

Keywords: Particle swarm algorithm, Inertial weight, Nonlinear decrease.

Abstract. The standard particle swarm optimization algorithm introduces the inertia weight $w$, which becomes a calculation method for finding the extremum of the function effectively. It is easy to converge fast. Nowadays, the linear decreasing dynamic, inertia weighting strategy is widely used. Although this improvement is very successful. However, the search process is a nonlinear and complex process. In order to better maintain the balance between global and local search capabilities, this paper proposes a nonlinear decreasing dynamic inertia weighting strategy based on the linear decreasing strategy, using Griewank, Rastrigin, Sphere. The simulation experiments of four standard test functions of JDSchaffer were carried out, and the inertia weights in the basic particle swarm optimization algorithm were compared with the nonlinear decrement of fixed weight $w = 0.95$, linear decreasing LDIW and Chen[6] exponential curve respectively. The improved nonlinear weight decrement strategy is superior to other algorithms in terms of convergence speed, convergence precision number of iterations.

Introduction

Based on the simulation of social behaviors such as bird foraging and human cognition, Kennedy and Eberhart [1] jointly proposed a global optimization algorithm, Particle Swarm Optimization (PSO), which is based on 1995. Evolutionary computing technology for swarm intelligence. The convergence speed of particle swarm optimization algorithm is slower when it is close to the most advantageous region. The particle swarm optimization algorithm has a faster convergence rate in the early stage, but in the late iteration, the improvement effect is not very satisfactory. This is mainly due to the convergence of the algorithm to the local pole. Small, lack of effective mechanisms to make the algorithm avoid the minimum point. In response to the above limitations, research scholars have made corresponding improvements. For example, in 1998, YHShi [4] proposed an improved particle swarm optimization algorithm with inertia weight–linear diminishing weighting strategy (LDIW); Li Huirong et al. [2] The index proposes a particle swarm optimization algorithm based on nonlinear decreasing inertia weighting strategy; Chen Guimin et al. [3] proposed nonlinear inertia weight reduction strategies such as exponential curve, open downward parabola and open upward parabola, etc. It is an improvement of the basic particle swarm optimization algorithm. Based on the previous research, this paper proposes a logarithmic nonlinear decreasing dynamic weighting strategy, and simultaneously analyzes and compares several other improved particle swarm optimization algorithms.

Basic Particle Swarm Optimization Algorithm

Suppose that in a target space with dimension $d$, the number of particle swarms is $n$, then the position and velocity of the $i$-th particle in the particle swarm can be represented by a $d$-dimensional vector, such as (2-1) and (2-2) shown:

$$X_i = (x_{i1}, x_{i2}, ..., x_{id})$$  

(2-1)
The optimal position that the \( i \) particle has searched so far is called the individual extremum, as shown in (2-3):

\[
P_{\text{best}} = (p_{i1}, p_{i2}, \ldots, p_{id})
\]

The optimal position that the entire particle swarm has searched so far is the global extremum, as shown in (2-4):

\[
g_{\text{best}} = (g_1, g_2, \ldots, g_d)
\]

When two optimal values are found, the particles update their speed and position according to equations (2-5) and (2-6):

\[
V_y(t + 1) = V_y(t) + c_1 r_1 [V_y(t) - X_y(t)] + c_2 r_2 [P_y(t) - X_y(t)]
\]

\[
X_y(t + 1) = X_y(t) + V_y(t + 1)
\]

Equations (2-5) and (2-6) form the basic particle swarm algorithm, where: \( i=1, 2 \ldots n, j=1, 2 \ldots d, n \) represents The number of particles in the particle swarm, and \( d \) represents the dimension of the target space. \( c_1 \) and \( c_2 \) indicate that the learning factor is usually set to \( c_1 = c_2 ; r_1 \) and \( r_2 \) are uniform random numbers and independent of each other, the value range is \([0, 1]\); \( V_y \) is the velocity of the particle, \( v_y \in [-v_{\text{max}}, v_{\text{max}}] \), \( v_{\text{max}} \) is a constant, and the magnitude of the velocity is determined by the objective function. The nature is determined.

**Linear Decreasing Inertia Weight Strategy**

In order to better control the development and detection capabilities of particle, YHShi introduces the inertia weight \( w \) into (2-5). Because the algorithm can guarantee better convergence, it is considered as a standard particle swarm algorithm. Its evolution process is as shown in equations (3-1) and (3-2):

\[
V_y(t + 1) = wV_y(t) + c_1 r_1 [V_y(t) - X_y(t)] + c_2 r_2 [P_y(t) - X_y(t)]
\]

\[
X_y(t + 1) = X_y(t) + V_y(t + 1)
\]

In addition, \( w \) can be dynamically adjusted during the search process. The more dynamic inertia weights are the linear decrement weights (LDIW) strategy proposed by Shi, that is, the linear \( w \) value is reduced linearly during the iterative process. Such as (3-3):

\[
w = w_i - (w_i - w_e) \frac{t}{T_{\text{max}}}
\]

where: \( w_i \) is the initial inertia weight, \( w_e \) is the end value of the maximum allowed number of iterations; \( T_{\text{max}} \) is the maximum number of evolutions; \( t \) is the current number of iterations, \( w_i = 0.95, w_e = 0.4 \) in most problem solving.

**Improved Nonlinear Decreasing Inertia Weight Strategy**

From basic particle algorithm to standard particle swarm optimization algorithm, researchers have done a lot of research. The proposed linear weighting strategy is widely used in various engineering multi-objective optimization problems. The optimized result graph of this strategy is simple. Intuitive and reflecting good performance, is this improvement the best declining strategy? Based on this problem, it is also found that the search process is a nonlinear and complex process.
Therefore, this paper proposes a nonlinear decrement strategy and tries to enhance the local and global search ability. The formula is (4-1):

$$w = \left[ w_s - f_i (w_e - w_s) \log \frac{t}{t_{\text{max}}} \right] / f_2$$

(4-1)

Among them, $w_s$, $w_e$, $T_{\text{max}}$ and $t$ have the same meaning as (3-3). This function represents a logarithmic function curve. $f_i$ and $f_2$ are adjustment factors. The purpose is to control the rate of change of $w$ so that its range satisfies $[0.049, 0.9]$. The improved strategy is used to maintain the balance between global search and local search. As the iteration progresses, the inertia weight decreases from 0.95 to 0.4, and the inertia weight $w$ takes a larger value in the algorithm of the iterative early iteration, thus avoiding the algorithm falling into local The extreme value maintains a strong global search ability; at the same time, a smaller inertia weight is taken in the late iteration to enhance the local search ability and speed up the convergence of the algorithm.

**Experimental Simulation Analysis**

This paper uses 4 classic test function (such as Table 1) to test these algorithms, except that the global minimum value of the JDSchaffer test function is $-1$, the global minimum values of the other 3 test functions are 0. The number of particles is 30, the dimension is 10, $f_i = 0.5$, $f_2 = 2.2$, $c_1 = c_2 = 2$, $w_s = 0.95$, $w_e = 0.4$. The ending condition is that the number of iterations exceeds 1000.

| Function     | Function expression | Dimension(d) | Search range     | Maximum speed |
|--------------|---------------------|--------------|------------------|---------------|
| **Griewank** | $\frac{1}{4000} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{n} \cos(x_i) + 1$ | 10           | $(-600, 600)^d$ | 600           |
| **Rastrigin** | $\sum_{i=1}^{d} [x_i^2 - \cos(2\pi x_i) + 10]$ | 10           | $(-10, 10)^d$   | 10            |
| **Sphere**   | $\sum_{i=1}^{d} x_i^2$ | 10           | $(-100, 100)^d$ | 100           |
| **J.D.Schaffer** | $\sin^2 \sqrt{x_1^2 + x_2^2 - 0.5} / (1 + 0.001 \times (x_1^2 + x_2^2)^2)$ | 10           | $(-10, 10)^d$   | 10            |

Table 1. Four standard test functions.

Figure 2. Griewank function.  
Figure 3. Rastrigrin function.
Result Analysis

Figure 2 shows that the convergence accuracy of the fixed weight is the worst and the convergence condition is not reached when the number of iterations is 1000. The inertia weight does not reach the convergence condition. Chen and the paper look at the overall optimization result, but this paper needs the number of iterations is small, which improves the speed and efficiency of convergence. Figure 3 shows that the line with fixed weight has the disadvantage of premature convergence, prematurely stops at the local best, and the convergence accuracy is far from the set accuracy. When the iteration number is 1000 times, the convergence condition is not reached. The convergence performance proposed in this paper is the best, followed by Chen, and the linear decrement is the worst. From Figure 4, we can see that the four algorithms are ideal, but the convergence of fixed weights. The speed is still poor, and the effect of introducing the inertia weight is better than that of the inertia weight; from Fig. 5, it can be seen that the four algorithms are closer to the set convergence precision when iterating to a certain number of times, and the line of fixed weight needs The number of iterations is the highest compared to the other three, and the overall optimization performance is still the worst, and the algorithm proposed in this paper still converges the fastest, the number of iterations is about 56, and the other 2 Algorithm in the intermediate state. In summary, the proposed nonlinear convergence rate is the fastest, the number of iterations required is the least, the convergence accuracy can be close to the algorithm, and the algorithm is not premature, which effectively improves the overall optimization performance of the algorithm.

Conclusion

In this paper, based on the research and analysis of the inertia weight of PSO particle swarm optimization algorithm, a nonlinear decreasing dynamic inertia weighting strategy is proposed. The algorithm sets a large weight in the early iteration, which is beneficial to jump out of local optimum; Weight, which is conducive to local optimization and speed up convergence. Through experiments on four standard test functions, we can see that the logarithmic nonlinear strategy proposed in this paper has faster convergence speed, higher convergence precision and fewer iterations, which better reflects the global search ability.

References

[1] Kennedy J., Eberhart R.C. Particle swarm optimization. IEEE International Conference on Neural Networks, 1995: 1942-1948.
[2] Li Huirong, Gao Yuelin, Li Jimin. A Particle Swarm Optimization Algorithm for Nonlinear Decreasing Inertia Weight Strategy [J]. Journal of Shangluo University, 2007(04): 16-20.
[3] Chen Guimin, Jia Jianyuan, Han Qi. Research on the Inertia Weight Reduction Strategy of Particle Swarm Optimization Algorithm [J]. Journal of Xi'an Jiaotong University, 2006(01): 53-56+61.
[4] Shi Y., Eberhart R.C. A modified particle swarm optimizer. Proceedings of the IEEE Conference on Evolutionary Computation, 1998: 69-73.

[5] Bao Ziyang. Intelligent optimization algorithm and its MATLAB example. Beijing: Publishing House of Electronics Industry, 2016.