Journal of Information & Computational Science 10:6 (2013) 1759–1766 April 10, 2013
Available at http://www.joics.com

# Crossover Particle Swarm Optimization with Incremental Learning ⋆

Jinyin Chen, Haibo Xu*, Haixia Lv

*College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China*

**Abstract**

Particle Swarm Optimization (PSO) algorithm is easy to fall into premature local convergence. In this paper, inheriting the crossover particle swarm optimization combined with the crossover operator, we proposed Crossover Particle Swarm Optimization with Incremental Learning (ILCPSO). ILCPSO builds two levels to overcome the local convergence and obtain the optimal solution. Firstly we introduce cross-operation into PSO, which exchanges good "genes" of particles in population according to certain crossover probability. It takes full advantage of information of the particle swarm to get the global optimal solution. Followed by the introduction of incremental learning, we add two sets of particles in prior to choose their own "first teacher", then according to the diversity of the population use two different learning methods to update their knowledge again. After competition with the respective "first teacher" the fittest will be survival, in order to ensure that the population size will not expand. Final validation as opposed to the crossover particle swarm optimization algorithm, the complexity of the algorithm has not increased significantly when the performance is greatly improved.

*Keywords*: Incremental Learning; Crossover Operation; Local Convergence; Multi-peak Optimization; Population Diversity

## 1 Introduction

Particle Swarm Optimization (PSO) was firstly brought up by Kennedy and Eberhart in 1995 [1], which was inspired by birds swarm searching for food in unknown area. Like Genetic Algorithm (GA), PSO is an optimization tool based on iteration. Compared with GA, PSO is easier to implement which has less parameter setting limitation and faster optimization speed. However PSO is easily got local optima instead of global one which would influence its performance and its further application. Li Yong etc. [2] brought up Lagrange multiplier method to improve standard PSO's capability of handling constrained optimization with decrease of optimizing time cost. Wang Hong etc. [3] adopted dynamic iteration weight strategy during several period of evolution, which guaranteed convergence speed obviously. Another paper [4] took advantage of artificial immune algorithm for optimization precision of PSO. Genetic operations have adopted for PSO

⋆Project supported by the Nature Science Foundation of Zhejiang Province (No. Y1100378).
*Corresponding author.
*Email address:* chenjinyin@163.com (Jinyin Chen).


DOI: 10.12733/jics20101662

in several ways. Mutation operation has been applied for increase new random particles in solution space for PSO. A novel PSO with adaptive mutation [5] was developed to increase possibility of searching unknown area of solution space. Crossover operation was also cited for PSO in paper [1], where two better particles selected for crossover operation to generate new particles. Theory and simulation results testified that crossover PSO has better convergence rate and optima in solving single-peak and multiple peaks optimization problems compared with other recently improved PSOs. In this paper, inspired from social incremental learning strategy's successful application in GA, we designed a crossover PSO combined with incremental learning algorithm called ILCPSO. In Section 2, we introduce how crossover PSO works and how incremental learning algorithm would possible has better performance than crossover PSO. Section 3 brought up main idea of ILCPSO, its flowing charts and main steps. Section 4 and 5 would focus on performance evaluation of ILCPSO compared with several classical versions of PSO algorithms. Finally we come to conclusion in Section 6.

## 2 Related Work

We have brought up Crossover PSO firstly in paper [1]. In traditional PSO, global optima are found by particles fly through solution space. Firstly they are initialized randomly in solution space and best particles are selected according to their fitness calculated by objections. And then the particles fly according to certain laws. Each particle adjusts its velocity vector, based on its momentum and the influence of its best solution *pbest* and the global best solution *gbest*. The original PSO updating equation are as follows.

$$V_i^{k+1} = \omega \times V_i^k + c_1 \times rand_1^k \times (pbest_i^k - X_i^k) + c_2 \times rand_2^k \times (gbest_i^k - X_i^k) \tag{1}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{2}$$

where $V$ is the velocity vector, $X$ is the particle position vector, $k$ is the $kth$ iteration, $i$ is the particle's ID. $c_1$ and $c_2$ are the positive acceleration constants. $rand$ is a random number between 0 and 1, and $w$ is the inertia weight.

While particles population easily loses diversity in traditional PSO, crossover operation could maintain population diversity, the main steps of crossover PSO are as follows.

**Step 1.** Initialize a group of particles including random position and velocity; and also calculate fitness of the initial swarm.

**Step 2.** Update position and velocity by Eq. (1) and (2). Evaluate each particle's fitness.

**Step 3.** Select two particles $p_1$ and $p_2$ as parents through tournament selection algorithm, where $Tournament$=5.

**Step 4.** Calculate crossover probability and in a certain probability select a crossover point among length $[0, maxLength]$ at random, and generate new particles $s_1$ and $s_2$. Evaluate fitness of $s_1$ and $s_2$. If fitness of $s_1$ or $s_2$ is better than old particles, then replace $p_1$ and $p_2$ with $s_1$ and $s_2$.

**Step 5.** If all particles have been updated then go to Step 6, or transfer to Step 3.

**Step 6.** According to fitness, select *pbest* and global optimum *gbest*.

**Step 7.** If the iteration achieve the maximum or the global optimum is got then end algorithm, or transfer to Step 3.

Incremental learning is a style of learning where the learner updates its model of the environment whenever a new significant experience becomes available. Compared to non-incremental or batch learning, incremental learning has the advantages of being more widely applicable and reactive. For example, it can be applied to situations where input data come only in sequence and a timely updating model is crucial for actions.

# 3 ILCPSO

## 3.1 Main Idea

Due to the inherent lack of characteristic in PSO, although each particle on memory function keeps the individual optimal solution of *pbest* and the whole population keeps the information of *pbest* in the process of solving, the potential of the optimal solution will also could be lost when parts of the particles' original advantages are lost because of the particle's information's all update in each generation. However, the crossover genetic algorithm is a good solution to the problem of the excellent chromosome's loss, then PSO combined with the cross operation therefore has a strong global search ability.

Unfortunately, the improved algorithm is also easy gotten into the local optimum, so we proposed crossover particle swarm optimization with incremental learning, which uses the selected operation to randomly select the relatively inferior particle $i$ as the new particle $i_1$'s "first teacher" after the cross operation. New particles completely inherit their teacher's existing knowledge and then we can calculate the diversity of population. If the diversity is less than threshold, it has the danger of gotten into local convergence because of the population is gathered around the *gbest*. Thus, particle $i_1$ enters into the pioneering learning state, begins to explore the open area and goes away from the former *gbest*. Or particle $i_1$ enters into the depth learning state, and goes to the model particle *gbest* and makes the population become quick-convergent.

In addition, we increase a new particle $i_2$ whose "first teacher" is $i_1$ as $i_1$' competitor and use a relative different learning mechanism on it. If the diversity is less than threshold $\lambda$, particle $i_2$ enters into the depth learning state, goes to the *gbest* and looks for possible bettor optimal solution between *gbest*, or otherwise it starts to self-study and begins to update the own knowledge according to *gbest*'s expected value and the optimal solution position's expectations judged within feasible search area. In the same study period, diversity's statement will not be changed. So if diversity is larger than $\lambda$, many groups of particles $i_2$ start to go for self-study and are gathered to the same position by the same study method which makes the population lose the variety. Consequently, we bring in a disturbance learning factor in $i_2$'s self-study to make $i_2$ produce a difference.

The increasing population will bring in operation time of the algorithm, and then we can take selection mechanism to overcome it, that we compare new particles after self-study with their "first teacher" and retain the superior while eliminating the inferior to keep the population scale invariance. Specific flow diagram is as shown in Fig. 1.

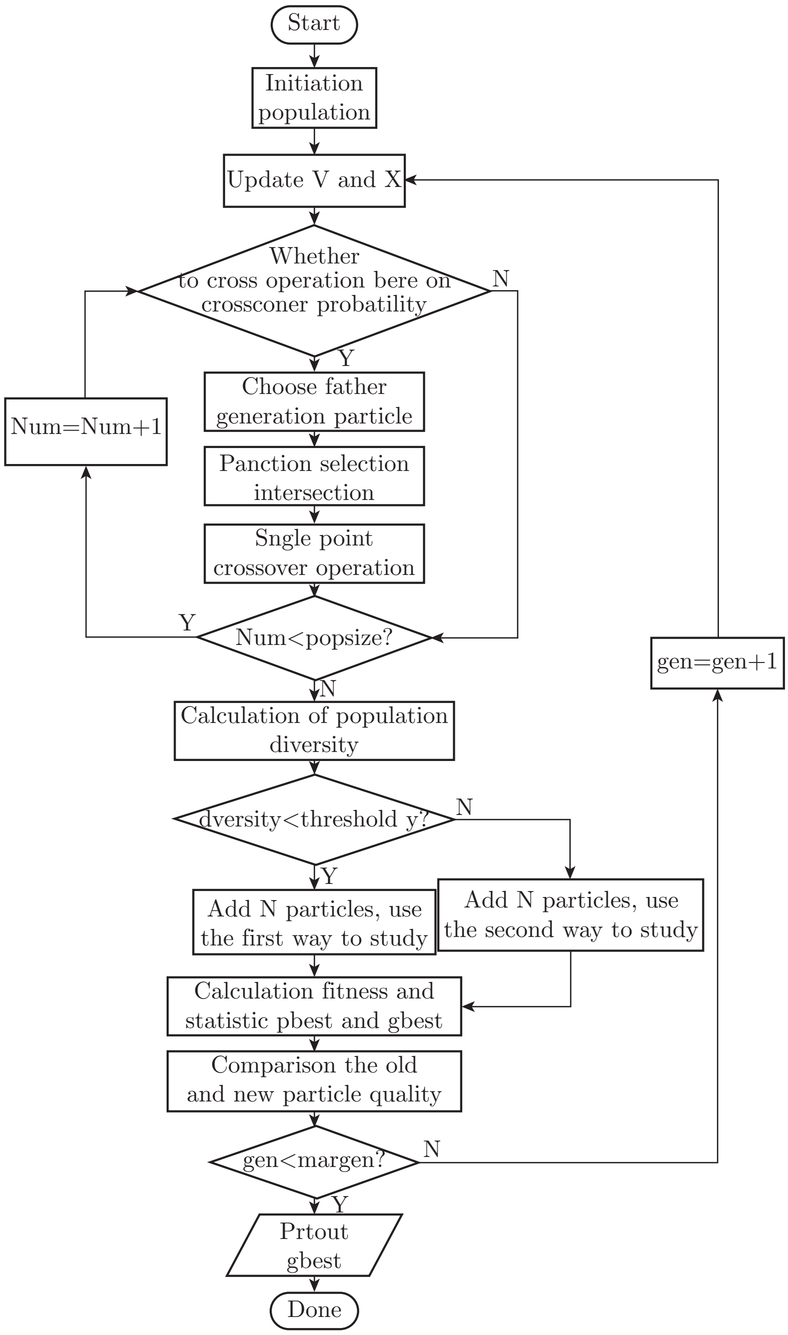


Fig. 1: Particles move toward so-far found global best optima instead of global best because of local trap

If diversity is less than $\lambda$, we choose the first method to study, that the particles $i_1$ begin to study as shown in Eq. (3), the particles $i_2$ begin to study as shown in Eq. (4); In contrast, we choose the second method to study, that the particles $i_1$ begin to study as shown in Eq. (3), the particles $i_2$ begin to study as shown in Eq. (5).

$$X_{st}(t) = \begin{cases} X_{min} + r \times (X_{max} - X_{min}) \times 0.25 & \text{when } |X_{max} - X_{gbest}| < |X_{min} - X_{gbest}| \\ X_{max} - r \times (X_{max} - X_{min}) \times 0.25 & \text{when } |X_{max} - X_{gbest}| > |X_{min} - X_{gbest}| \end{cases} \tag{3}$$

$$X_{st}(t) = X_{ts}(t) + r \times (X_{gbest} - X_{ts}(t)) \tag{4}$$

$$X_{st}(t) = X_{gbest} \times 0.25 + (X_{max} + X_{min}) \times 0.5 + r \times (X_{gbest} - X_{ts}(t)) \times 0.1 \tag{5}$$

where $X_{st}(t)$ means the increased particles' position, $X_{ts}(t)$ means the increased particles's teacher's position, $r$ is a random value between 0 and 1, every dimension uses different random value to represent different learning strength.$X_{gbest}$ is the current global population's optimal place. $X_{max}$ is the function's upper limit while $X_{min}$ is the lower limit. That $r \times (X_{gbest} - X_{ts}(t)) \times 0.1$ means the disturbance learning factor when in self-study(every dimension uses different random value to abound the new particles' variety.

## 3.2 The Analysis of the Algorithm's Convergence

If diversity is less than $\lambda$, particles $i_1$ will be divergent and particles $i_2$ will be convergent as iterations are big enough according to Eq. (3) and (4). It means that when $\forall \varepsilon > 0$, $\exists t > N$, $\lim(X_{gbest} - X_i(t)) < \varepsilon$, $\lim X_{i2}(t) = X_{gbest}$.

Similarly we can confirm that particles $i_1$ and $i_2$ will be both convergent as iterations are big enough according to Eq. (3) and (4) when diversity is more than $\lambda$. At this time,

$$\lim X_{i1}(t) = X_{gbest} \tag{6}$$

$$\lim X_{i2}(t) = 0.25 \times X_{gbest} + 0.5 \times (X_{max} + X_{min}) \tag{7}$$

So whatever statement the current diversity is in population, there will be some parts of particles in the global optimal solution.

## 3.3 The Analysis of the Algorithm's Complexity

ILCPSO's complexity in this paper mainly includes four parameters that are $M$ (population size), $N$ (variable number), $t$ (incremental learning interval algebra), $L$ (the study of particle number). The order of operation is as follows. Firstly we initialize $MN$, the velocity of particles and update the position $2MN$. Secondly, we calculate $MN$'s adaptive value which is $0.5M(2N + 2N + 2N)$ in crossover operation. Thirdly, we go for the incremental learning $L(N + N)/t$, and fourthly we calculate the variety $2MN$. Then we can get the total numbers $P_n$ which is averagely calculated from the $nth$ iteration algorithm and it should meet as follows:

$$P_n \leq MN + 2MN + MN + 0.5M(2N + 2N + 2N) + L \times (M + N)/t + 2MN = 9MN + 2LN/t \tag{8}$$

Moreover, the parameter $M$ is being valued far greater than $N$, and $L = M$, so we can infer that ILCPSO's time complexity has the maximum $O(M^2)$.

## 3.4 Experimental Analysis on Algorithm

Every time the experiment is run 10 times and 1000 generations (Besides F3 is run 300 generations), then we can obtain the best adaptive value, the worst adaptive value, and their mean value and variance which are utilized to compare their properties.

As the selecting operation is just to choose a relative best or worst particle, we can select a proper group of comparative particles whose numbers are not too much to save operation time. And in this paper, we let the number of particles be 3.

In the crossover operator, we use the crossover probability model which function is $pc = (pcmax - pcmin) * (maxgen - gen)/(maxgen) + pcmin$, $pcmax = 0.9$, $pcmin = 0.6$. And $w$ could be calculated by formula $\omega = \omega_{max} - (\omega_{max} - \omega_{min}) \times gen/gen_{max}$, where $\omega_{max} = 0.9$, $\omega_{min} = 0.6$, $c_1 = c_2 = 2$, $Vmax = 1$, and particle population size $pNum = 200$.

Through the optimization experiment among the 10 classical optimization test function, we can compare ILCPSO with the other four algorithm standard which are PSO [1], LDWPSO [3], QPSO and crossPSO [3]. The experimental results listed in Table 1 shows that whether crossPSO or ILCPSO, the consumption time of them are much higher than other algorithms and need more about 70 percent to 100 percent consumption time than standard PSO.

Table 1: Average operation time about CPU consumption in respective algorithms (s)

| Test function (dimension) | PSO | LDWPSO | QPSO | crossPSO | ILCPSO |
|---|---|---|---|---|---|
| F1(30) | 1.8437 | 1.8333 | 2.0316 | 3.3923 | 3.6579 |
| F2(30) | 2.4278 | 2.2741 | 2.6186 | 4.4808 | 4.6044 |
| F3(10) | 0.1965 | 0.1961 | 0.2265 | 0.4754 | 0.4752 |
| F4(30) | 1.7726 | 1.7857 | 2.0602 | 3.3533 | 3.4376 |
| F5(30) | 1.2828 | 1.2795 | 1.5162 | 2.3949 | 2.5108 |
| F6(30) | 2.1727 | 2.1384 | 2.3718 | 3.902 | 4.0048 |
| F7(30) | 1.702 | 1.7488 | 1.8183 | 2.9585 | 3.1093 |
| F8(30) | 3.7486 | 3.5923 | 3.8784 | 6.482 | 6.9672 |
| F9(30) | 1.8609 | 1.8144 | 2.0886 | 3.4271 | 3.7433 |
| F10(2) | 0.1848 | 0.1873 | 0.2122 | 0.8451 | 0.8529 |

The experimental results listed in Table 2 show that ILCPSO will improve the convergence precision much greater than crossPSO in the case of consumption time's increase is not much; ILCPSO can solve multi-modal optimization problem very well, and the property is also much better than crossPSO.

Table 2: Test result about F1∼F10

| Test function (dimension) | PSO | QPSO | CrossPSO | LDWPSO | ILCPSO |
|---|---|---|---|---|---|
| F1 Sphere(30) | 1.15356 | 1.31E-45 | 1.97E-53 | 1.7221E-12 | 2.55E-268 |
| F2 Ackley(30) | 2.6572 | 1.84081 | 1.84081 | 1.84081 | 1.84081 |
| F3 Trid(10) | −209.954 | −210 | −210 | −210 | −210 |
| F4 Sum Squares(30) | 33.7851 | 4.44E-46 | 5.97E-53 | 7.59E-11 | 2.00E-260 |
| F5 Rastrigin(30) | 161.217 | 5.96975 | 22.884 | 26.8639 | 0 |
| F6 Griewank(30) | 2.6376E-4 | 1.38E-48 | 1.52E-46 | 1.88E-09 | 6.95E-263 |
| F7 Schwefel(30) | 5325.18 | 947.507 | 1658.16 | 8023.62 | 1875.31 |
| F8 Rosenbrock(30) | 184.49 | 6.88845 | 1.59E-05 | 24.3201 | 9.97E-08 |
| F9 Zakharov(30) | 2.484 | 21.324 | 1.58E-13 | 1.29666E-2 | 2.11E-13 |
| F10 Schaffers(2) | 3.03E-11 | 0 | 0 | 0 | 0 |

Fig. 2 is the convergence diagram in each function for each algorithm, which indicate that ILCPSO has a greatly fast convergence speed compared with other algorithms.

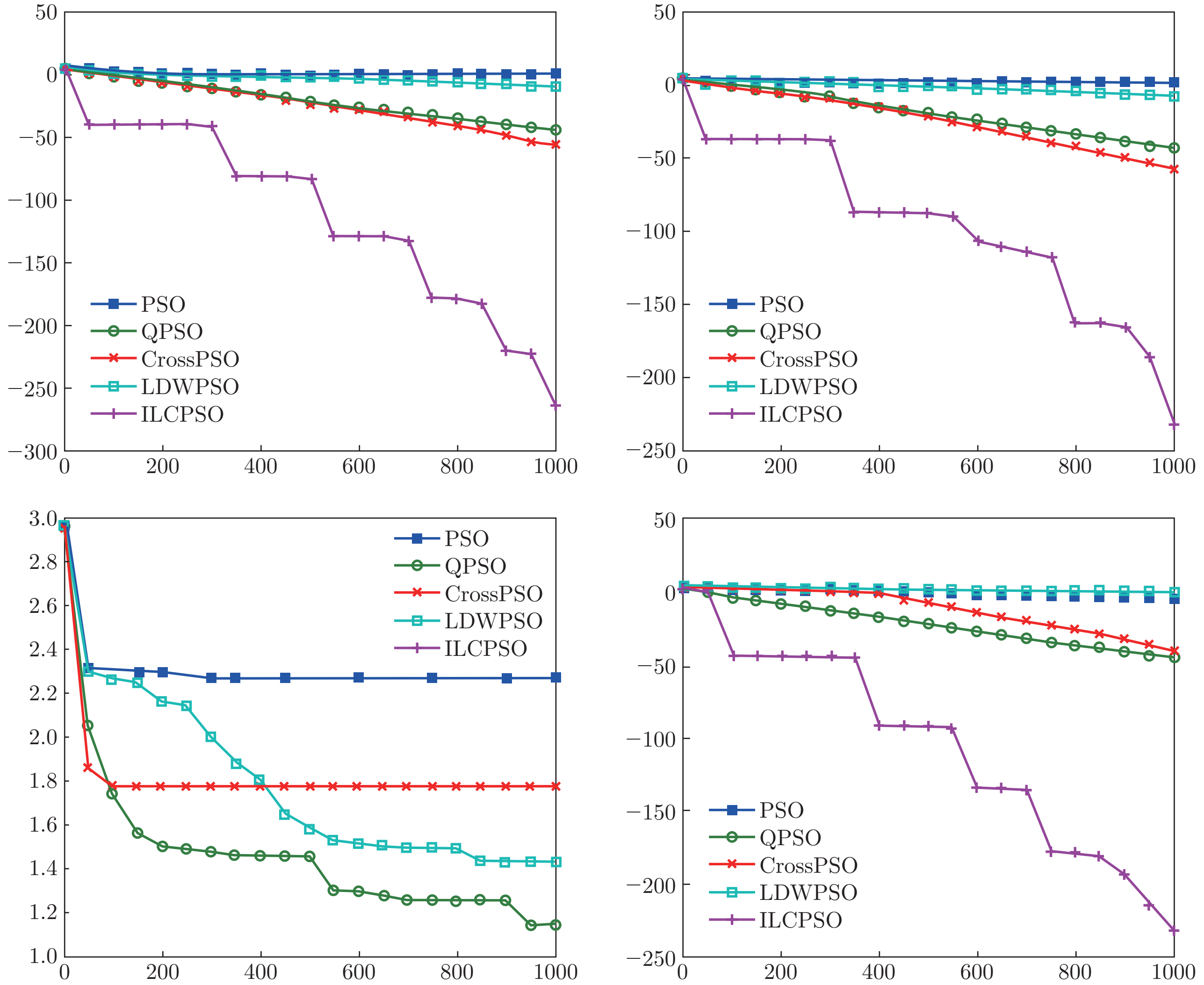


Fig. 2: Convergence os PSOs in optimizing test functions

# 4 Conclusions

In this paper, which is based on the crossover particle swarm optimization, proposed crossover particle swarm optimization with incremental learning and meanwhile proved the convergence of incremental learning method. The results indicate that ILCPSO can make convergence precision and convergence speed greatly improved when overcoming the local convergence. As a result, ILCPSO can improve the algorithm's performance level and effectively solve the unimodal and multimodal optimization problems.

# References

[1] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multi-dimensional complex space, IEEE Trans. Evol. Comput., Vol. 6, Feb. 2002, 58-73

[2] Yong Li, Jianjun Wang, Lihua Cao, Simulated annealing particle swarm optimization algorithm of optimal load dispatch in power plant, Proceedings of the CSU-EPSA, 6(3), 2011, 40-44

[3] Jinyin Chen, Dongyong Yang, Jin Lu, Self-adaptive crossover particle swarm optimization based on penalty mechanism, Computer Science, 37(4), 2010, 249-254

[4] Hongtao Wang, Yan Ren, Particle swarm of timisation algorithm based on modified inertia weight, Computer Applications and Software, 10(10), 2011, 23-30

[5] Pingping Xia, Taizhi Lv, Yanfeng Jia, Performance analysis of immune particle swarm optimization algorithm, Journal of Guizhou University (Natural Sciences), 28(5), 2011, 104-107