IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 6, November 2010
ISSN (Online): 1694-0814
www.IJCSI.org

A Modified Algorithm of Bare Bones Particle Swarm Optimization

Horng-I Hsieh[1] and Tian-Shyug Lee[2*]

[1] Graduate Institute of Business Administration, Fu-Jen Catholic University
Hsin-Chuang, Taipei County 24205, Taiwan, ROC

[2] Department of Business Administration, Fu-Jen Catholic University
Hsin-Chuang, Taipei County 24205, Taiwan, ROC

Abstract

Bare bones particle swarm optimization (PSO) greatly simplifies the particles swarm by stripping away the velocity rule, but performance seems not good as canonical one in some test problems. Some studies try to replace the sampling distribution to improve the performance, but there are some problems in the algorithm itself. This paper proposes a modified algorithm to solve these problems. In addition to some benchmark test functions, we also conducted an application of real-world time series forecasting with support vector regression to evaluate the performance of the proposed PSO algorithm. The results indicate that the modified bare bones particle swarm optimization can be an efficient alternative due to the smaller confidence intervals and fast convergence characteristics.

Keywords: *Heuristic Optimization, Particle Swarm Optimization, Bare Bones PSO, Support Vector Regression, Time Series Forecasting.*

***Corresponding author.**

1. Introduction

Particle swarm optimization (PSO) is a population-based heuristic method developed by Kennedy and Eberhart in 1995 [1]. The PSO algorithm is inspired by the collective motion of biological organisms, such as bird flocking and fish schooling, to simulate the seeking behavior to a food source. A PSO algorithm is initialized with a population of random particles treated as a point in D-dimensional search space. To find the optimum solution, each particle adjusts the direction through the best experience which it has found (pbest) and the best experience been found by all other members (gbest). Therefore, the particles fly around in a multidimensional space towards the better area over the search process.

The PSO system initially has a population of random solutions and then searches for optimum solution by updating process. Each particle consists of three vectors: the position for ith individual particle $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, the best previous position that the ith particle has found $\mathbf{p}_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, and its velocity $\mathbf{v}_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. The performance of each particle is measured using a fitness function varying from problem in hand. During the iterative procedure, the particle's velocity and position are updated by

$$v_{iD}^{new} = \chi \begin{pmatrix} v_{iD}^{old} + c_1 \times rnd() \times (p_{iD} - x_{iD}^{old}) \\ + c_2 \times Rnd() \times (p_{gD} - x_{iD}^{old}) \end{pmatrix} \quad (1)$$

$$x_{iD}^{new} = x_{iD}^{old} + v_{iD}^{new} \quad (2)$$

where c_1 and c_2 are two positive acceleration constants, χ is a constriction factor, p_{iD} is the pbest of ith particle and p_{gD} is the gbest of the group, and *rnd()* and *Rnd()* are two random numbers uniformly generated from [0,1]. In a PSO system, particles change their positions at each time step until a relatively unchanging position has been encountered or a maximum number of iterations has been met.

Kennedy [2] proposed a new PSO where the usual velocity formula is removed and replaced with samples from a Gaussian distribution. The velocity-free bare bones (BB) PSO was inspired by the observation that histogram sampled by the canonical particle swarm is appeared to be normally distributed around $(p_{iD} + p_{gD}) / 2$, with a standard deviation of $| p_{iD} - p_{gD}|$. Many factors have been found to determine how successful the problem-solving process are often problem dependent. The bare bones PSO using information drew from a Gaussian distribution greatly simplifies the particles swarm algorithm, but the performance in Gaussian version is not as good as canonical PSO [3][4]. Some researchers [3][5] try to examine the bell shape distribution and replace the Gaussian random number generator by the appropriate one to reproduce the behavior of canonical algorithm, and improvements have been observed. However, the problem

of bare bones PSO might not be the replaced distribution, but the algorithm itself. The bare bones PSO might suffer from premature convergence or converge to a point neither global nor local optimum.

The aim of this paper is to propose a new algorithm to avoid the problem mentioned above. Furthermore, the performance of the proposed algorithm in some benchmark functions and a real-world application are investigated. The remainder of the paper is organized as follows: An overview and a closer examination on convergence behavior of bare bones PSO is given in Section 2. Section 3 provides the modified algorithm of bare bones particle swarm optimization. Benchmark functions to measure the performance of the different approaches are provided in Section 4. Section 5 presents the results of a real-world application and conclusions are drawn in Section 6.

2. Bare Bones PSO

There are several problems appear in the bare bones PSO. Firstly, the best particle in bare bones acts different from the canonical one. In a canonical particle swarm, particles change their positions at each time step. On the other hand, the best particle of the neighborhood in bare bones PSO simply stands in its best previous position due to the random number generator definition. If the other particles move too close to the best one in their neighborhood, the particles may converge on a point that is neither the global nor the local optimum. Consider a simplified situation shown in Figure 1, which with only two particles and one dimension. Since particle A is the best one, it will always stand in the same position as particle B moves to the right side without finding out any better result. If particle B flies into $[P_A, P_B]$ or $[0, P_A]$ but too close to particle A, the PSO system will become inactive. The particles will move very slowly in future iterations; even converge on a wrong position. Fortunately, as the number of particles grows, the probability that system becomes ineffective decreases. However, if the problem becomes complicated, determining an appropriate number of particles might be a difficult task because the growing size of swarm also increases the time of computation.

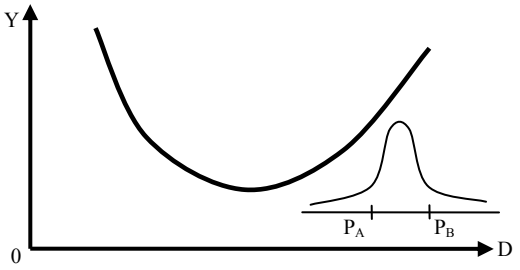

Fig. 1 A simplified particle swarm system.

Additionally, bare bones particle swarm might suffer from premature convergence. When the best particle locates in suboptimal position, it tends to mislead all the other particles to get stuck in this local optimum. Each particle in the neighborhood can fast approach to the best area within few iterations by making large step sizes even if the particles whose personal previous best position are far away from the global best position. Consider the simple situation depicted in Figure 2. All the particles move fast toward to the p_g inside a local optimum area. If each of the particles fails to hit the region of the global optimum, the whole system might lose exploration capability as the standard deviation cannot be back to large value.

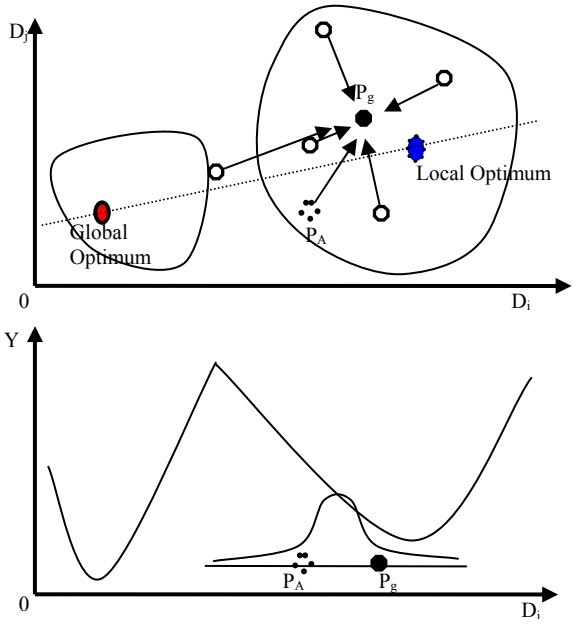

Fig. 2 All of particles are prematurely converging to local optimum.

3. A Modified Bare Bones PSO

As mentioned in the preceding section, the best particle without momentum might harm the performance of the whole system. Our strategy is to modify formula of standard deviation by adding a new parameter. If the particle is the best one in its neighborhood, the new standard deviation is computed as

$$| \delta \times p_{iD} - p_{gD} | = | \delta \times p_{gD} - p_{gD} |, \qquad (3)$$

where δ is a constant, which may be a number either larger or small than 1. The offset of the best position behaves similar to the momentum term in velocity update rule of canonical PSO. Table 1 shows the results of the proposed

standard deviation computing strategy with $\delta = 1.2$. As the results reveal, the new strategy can efficiently solve the above-mentioned problem.

Table 1: Mean results of De Jong after 50 trials of 3000 iterations

Dimensions / Methods		Number of particles			
		2	*3*	*4*	*5*
2	original	7372.01	4113.35	2016.80	299.57
	modified	2.6E-117	8.86E-226	0	0
5	original	19663.9	11706.5	5383.8	1324.67
	modified	4.93E-55	6.91E-103	5E-151	9.1E-189

Numbers that are less than 4.9407E-324 are rounded to 0.

Since the best particle might mislead all the other particles into local optimum very fast, one thought is to slow down the speed of the movement. Because of forcing the particles to reach the region close to the global optimum might reduce the chance of getting stuck in local optimum; a slower convergence by using a smaller step size in the earlier stage might be beneficial. On the other hand, if the particle inside one of a local optimum leading by the best particle also has the capability to reach the region near the global optimum, it is likely to escape from the inferior suboptimal. Thus, there are two strategies can deal with the premature convergence situation:

1. Offset the mean to force the particle not being very close to the best particle during the early stage.
2. Slow down the biased exploration by constraining the standard deviation and offsetting the mean together during the early stage.

Thus, the mean and standard deviation of the Gaussian distribution used to update the position in the early stage becomes

$$X_{iD} \sim N\left(\omega_1\left(\frac{P_{iD}+P_{gD}}{2}\right), \omega_2\left|P_{iD}-P_{gD}\right|\right) \quad (4)$$

where ω_1 and ω_2 are two constriction parameters which can be either a fix or dynamic value over the early stage. A series of experiments were conducted by using eight benchmark functions to investigate these ideas. All the test functions were implemented in 30 dimensions except for the two-dimensional Schaffer's f6 function. Their definitions and initial range are shown in Table 2. Twenty particles were used in the test presented here. Each experiment was implemented 100 times for 3000 iterations. All algorithms are initialized asymmetrically with the ranges as shown in Table 2.

Table 2: Benchmark functions

Name	Equation	D	Initialization
De Jong	$f(\mathbf{x}) = \sum_{i=1}^{D} x_i^2$	30	$(50,100)^D$
Schwefel 1.2	$f(\mathbf{x}) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	30	$(50,100)^D$
Schaffer's f6	$f(\mathbf{x}) = 0.5 + \frac{(\sin\sqrt{x^2+y^2}) - 0.5}{\left(1.0 + 0.001\left(x^2+y^2\right)\right)^2}$	2	$(50,100)^D$
Rosenbrock	$f(\mathbf{x}) = \sum_{i=1}^{D-1}\left(100\left(x_{i+1}-x_i^2\right)^2 + \left(x_i-1\right)^2\right)$	30	$(15,30)^D$
Rastrigrin	$f(\mathbf{x}) = \sum_{i=1}^{D}\left(x_i^2 - \cos\left(2\pi x_i\right) + 10\right)$	30	$(2.56,5.12)^D$
Schwefel 2.6	$f(\mathbf{x}) = -\sum_{i=1}^{D} x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	30	$(-500,-250)^D$
Griewank	$f(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$(300,600)^D$
Ackley	$f(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi x_i\right)\right) + 20 + e$	30	$(16,32)^D$

Table 3 summarizes the results. In this test, ω_1 and ω_2 were set as constants with a large and a small value, and the constriction was only conducted before the 100th iteration. After the adjustment in the early stage, the ω_1 and ω_2 were set back to 1 to ensure the convergence. Also, the position offset factor of the best particle δ was set to 1.2. Each setting has good performance in some test problems are shown in bold. Due to the generalization ability consideration, the third setting which ω_1 and ω_2 are set to 0.7 will be used thereafter. For convenience, the rest of this paper will use the term BBM (modified bare bones) as the abbreviation.

Table 3 Mean results of parameter test after 100 trials of 3000 iterations

	De Jong	Schwefel 1.2	Schaffer's f6	Rosenbrock
M 0.7	5.48E-58 (3.15E-57)	0.00258 (0.00710)	2.72E-03 (4.38E-03)	22.48945 (0.39626)
M 0.4	3.27E-112 (2.33E-111)	0.00450 (0.01344)	1.75E-03 (3.75E-03)	22.21035 (0.32638)
MS 0.7	7.23E-70 (2.89E-69)	0.00034 (0.00056)	0.00311 (4.56E-03)	22.08622 (0.38908)
MS 0.4	**1.47E-118** (1.10E-117)	**1.21E-06** (4.35E-06)	**1.94E-04** (1.37E-03)	**22.05447** (0.28647)
	Rastrigrin	Schwefel 2.6	Griewank	Ackley
M 0.7	0.74622 (3.15704)	-8358.08 (585.55)	0.00398 (0.01227)	1.70E-14 (4.75E-15)
M 0.4	**0** (0)	-7923.94 (1027.71)	**0** (0)	**9.95E-16** (6.09E-16)
MS 0.7	7.03436 (13.38337)	**-8693.92** (428.13)	**0** (0)	1.48E-14 (5.35E-15)
MS 0.4	**0** (0)	-7382.71 (1396.69)	**0** (0)	1.53E-15 (1.37E-15)

Note that M means only mean offset method is used, and MS means constrict the mean and standard deviation in the same time. Standard deviations are shown in parentheses. Numbers that are less than 4.9407E-324 are rounded to 0.

4. Experimental Results

This section compares the performance of BBM with BB and canonical PSO. The three coefficients of canonical PSO were set as $\chi = 0.7298$ and $c_1 = c_2 = 2.05$ [6]. Eight benchmark functions shown in Table 2 were used to compare the performance of BBM with those of other algorithms. Gbest, Ring, and Square topologies were tested for all algorithms. A swarm size of 20 was used in all experiments, and each experiment was run 100 times for 3000 iterations. Also, the algorithms were initialized asymmetrically and the ranges did not contain global optimum, which can be found in Table 2.

As seen in Table 4, BB shows comparable results with canonical PSO on some test functions, but has the worst result on Rosenbrock with all topologies. On the other hand, BBM shows the best results on 6 out of 8 functions across all topologies, and outperforms BB on 7 out of 8 functions when using the Square topology. Note that BBM is the only one able to find the global minimum to the Rastrigrin, Griewank and Ackley functions, and the best results with the smallest standard deviations to Rosenbrock function. Figure 3 shows the mean performance best over time with the Gbest topology. As seen in Figure 3, BBM achieved a faster reduction than BB on all of the test functions. In summary, BBM provides better results with smaller confidence intervals compared to BB, thus can be a competitive optimizer on these test functions.

Table 4: Mean results of eight test functions after 100 trials of 3000 iterations

	De Jong	Schwefel 1.2	Schaffer' s f6	Rosenbrock
Canonical Gbest	2.09E-23 (1.58E-22)	3.71E-16 (3.71E-15)	3.98E-03 (4.80E-03)	33.40252 (34.20939)
Ring	1.53E-23 (2.62E-23)	**5.62E-30** (3.19E-29)	2.60E-03 (4.30E-03)	57.52692 (51.60641)
Square	7.06E-30 (4.10E-29)	1.19E-28 (3.34E-28)	1.55E-03 (3.58E-03)	48.23838 (40.89987)
BB Gbest	6.53E-23 (6.53E-22)	0.00655 (0.01662)	5.44E-03 (4.85E-03)	45.26925 (38.83877)
Ring	1.52E-10 (1.08E-09)	0.00077 (0.00245)	1.30E-03 (3.14E-03)	93.73233 (90.37186)
Square	8.88E-20 (5.72E-19)	0.00381 (0.01617)	1.63E-03 (3.63E-03)	55.52461 (45.11918)
BBM Gbest	**7.23E-70** (2.89E-69)	0.00034 (0.00056)	0.00311 (4.56E-03)	**22.08622** (0.38908)
Ring	9.62E-48 (1.97E-47)	3.72E-05 (0.00017)	2.14E-03 (4.05E-03)	24.15358 (0.14746)
Square	3.85E-53 (1.36E-52)	0.00010 (0.00022)	**1.17E-03** (3.17E-03)	23.32601 (0.18634)
	Rastrigrin	**Schwefel 2.6**	**Griewank**	**Ackley**
Canonical Gbest	189.58859 (43.01228)	-8638.29 (259.66)	0.17473 (0.63463)	19.72528 (0.29512)
Ring	173.49893 (35.18645)	**-9068.70** (195.36)	0.00570 (0.00847)	19.75951 (0.08589)
Square	156.15821 (39.14282)	-8930.48 (222.24)	0.01568 (0.02112)	18.79553 (4.06082)
BB Gbest	147.89009 (29.15393)	-8982.94 (336.97)	0.04022 (0.07372)	18.22 (4.29)
Ring	178.03970 (30.28106)	-9032.27 (253.16)	0.00955 (0.01786)	19.37 (1.90)
Square	151.97253 (28.63182)	-8987.26 (268.43)	0.01192 (0.01435)	17.53 (5.28)
BBM Gbest	7.03436 (13.38337)	-8693.91 (428.13)	**0** (0)	1.48E-14 (5.35E-15)
Ring	**0** (0)	-8527.74 (432.08)	**0** (0)	1.01E-14 (3.03E-15)
Square	**0** (0)	-8745.25 (356.85)	**0** (0)	**9.81E-15** (3.09E-15)

5. Model Selections in Support Vector Regression

Support vector machine (SVM) is a novel neural network algorithm based on statistical learning theory [7]. With introduction of Vapnik's ε-insensitivity loss function, the regression model of SVMs, called support vector regression (SVR), has been receiving increasing attention to solve nonlinear regression problems. In the modeling of SVR, one of the key problems is how to select model parameters correctly, which plays an important role in good generalization performance. However, no general guidelines are available to choose the free parameters of an SVR model. This section demonstrates a financial time series forecasting problem by using PSO to search the optimal parameters of SVR model selections. In order to evaluate the performance of the proposed approach, the Nikkei 225 closing cash index is used as the illustrative example.

Because the real optimum is unknown, a 'grid-search' method is applied to find the best combination of parameters. The searching space of the three parameters was set in $[2^{-15},2^{15}]$ with a step size 0.5. The minimum root mean squared error (RMSE) 57.64 found by the gird-search method is considered as optimum in forecasting the Nikkei closing cash index. The task of PSO here is to find a set of parameters with acceptable even better accuracy. The number of dimension is equal to 3 as there are three free parameters in SVR. Each trial was randomly initialized in $[2^{-15},2^{15}]$. Because modeling SVR is a time-consuming task, 10 particles were used in this section, and each experiment was implemented 50 times for 200 iterations. Any trial reaches the criterion equal to 58.2164 was treated as a success case. Two constriction parameters in BBM were set to 0.7, and set back to 1 after 30 iterations.

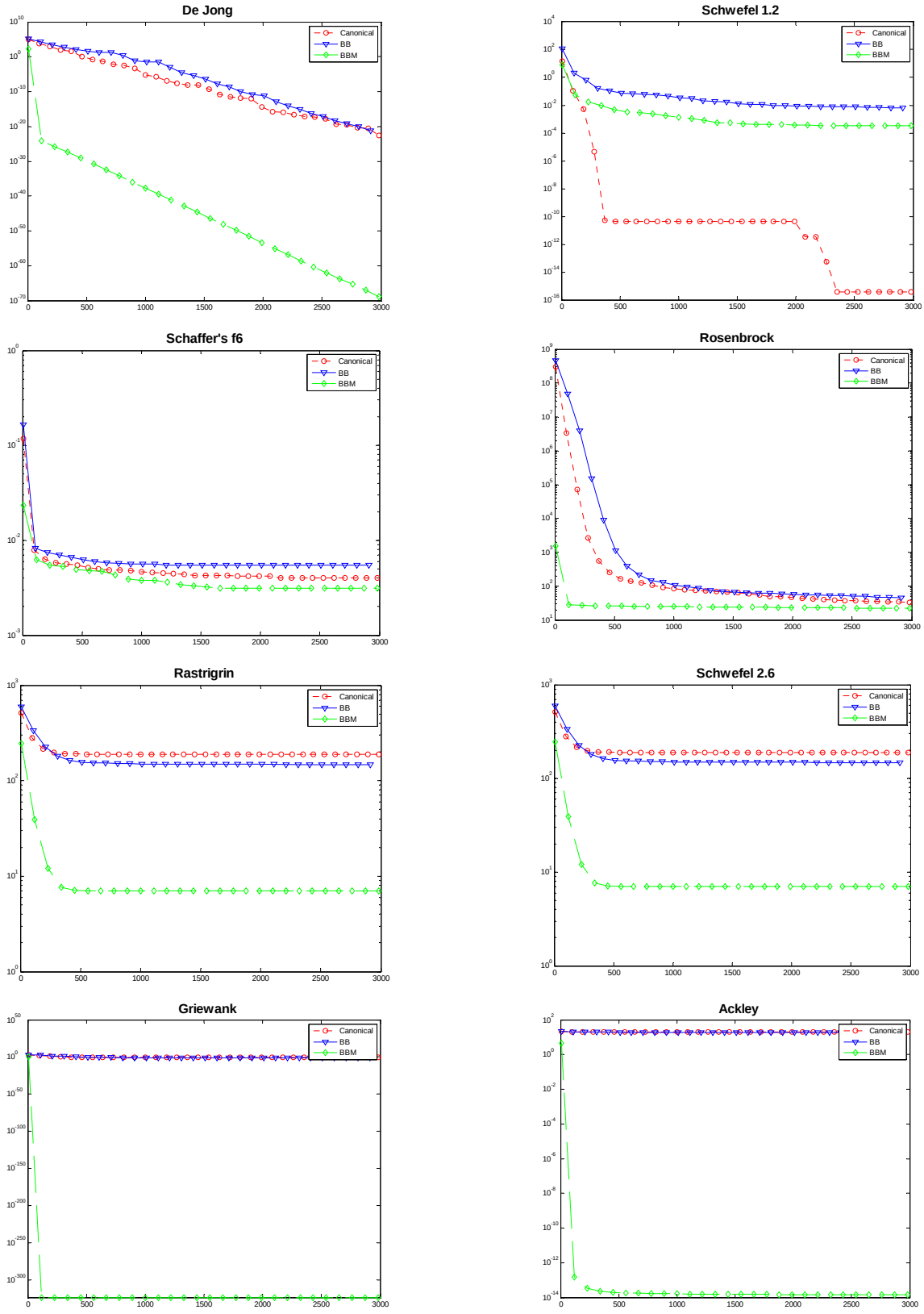

Fig. 3 Comparison between three algorithms with eight benchmark functions.

Table 5 shows the results obtained from three algorithms. All algorithms using the Ring topology return better results than the Gbest one. Canonical PSO with Ring topology is the best one with the smallest mean RMSE. In addition, BBM using Ring topology outperforms BB in lower RMSE and the smallest standard deviations. Figure 4 shows that BBM converges very fast with both Gbest and Ring topologies. Furthermore, the strategy that a coarse grid followed by a finer grid using in SVM with two parameters might not be applicable here, because there are three parameters in SVR, the better solution is not necessarily nearby another good solution. Thus, the grid-search might not easily find any better solution. On the other hand, improvement is still possible because the early stop criterion with 200 iterations was used here. The minimum found by three algorithms shows in Table 5 also prove its potential.

Table 5: SVR results of three algorithms

	Canonical		**BB**		**BBM**	
	Gbest	***Ring***	***Gbest***	***Ring***	***Gbest***	***Ring***
Mean	57.79 (0.54)	**57.47** (0.34)	57.71 (0.85)	57.61 (0.35)	57.86 (0.53)	57.56 (0.23)
Max	59.48	58.51	60.53	58.63	58.63	**58.40**
Min	**55.99**	56.42	54.48	56.74	**55.99**	57.27
Successful rate	86.00%	**96.00%**	80.00%	92.00%	56.00%	94.00%

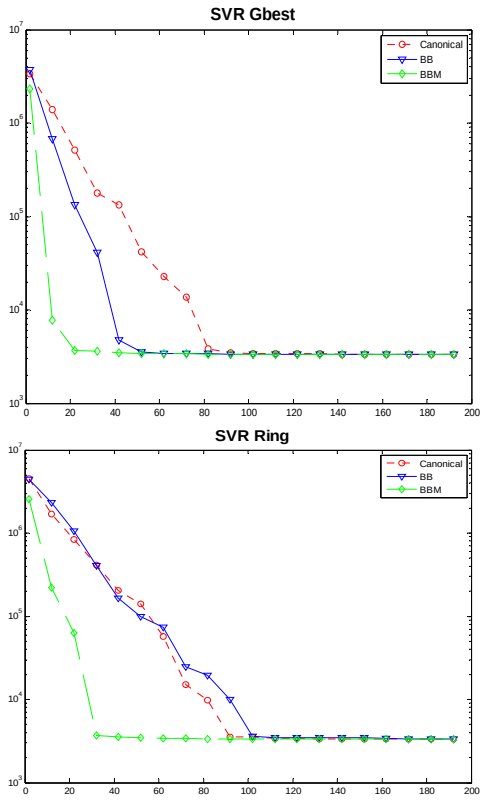

Fig. 4 SVR results of three algorithms.

6. Conclusions

This paper proposed a modified bare bones PSO by adding three extra parameters to correct the problems appears in original bare bones PSO. The modified algorithm shows advantages in better performance with smaller standard deviation and faster convergence characteristics. The restart strategy to deal with the problem converging to local optimum might harm than benefit the swarm. In Contrast, the offset strategy applied in early stage shows very good performance without harming the capability of convergence. Therefore the proposed algorithm can be a competitive optimizer.

Acknowledgments

This research was partially supported by the National Science Council of the Republic of China under Grant Number NSC 97-2221-E-030-011-MY2.

References

[1] J. Kennedy, and R.C. Eberhart, "Particle Swarm Optimization", in Proc. IEEE Intell. Conf. Neural Networks, 1995, Vol. IV, pp. 1942-1948.
[2] J. Kennedy, "Bare Bones Particle Swarms", in Proc. IEEE swarm Intell., 2003, pp. 80-87,.
[3] J. Kennedy, "Probability and Dynamics in the Particle Swarm", in Proc. IEEE Cong. Evolutionary Computation, 2004, Vol. 1, pp. 340-347.
[4] J. Kennedy, "Dynamic-probabilistic Particle Swarms", in Genetic Evolutionary Computation Conf., 2005, pp. 201-207.
[5] T.J. Richer, and T.M. Blackwell, "The Lévy Particle Swarm", in Proc. IEEE Cong. Evolutionary Computation, 2006, pp. 808-815.
[6] M. Clerc, and J. Kennedy, "The Particle Swarm: Explosion, Stability, and Convergence in a Multi-dimensional Complex Space", IEEE Trans. Evolutionary Computation, 2002, Vol. 6, pp. 58-73.
[7] V.N. Vapnik, The Nature of Statistical Learning Theory, 2nd ed., NY: Springer, 2000.