EVALUATIONS OF PARTICLE SWARM OPTIMIZERS FOR MULTIMODAL FUNCTION OPTIMIZATION

Kong Lingdong

School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

Email: KONG0129@e.ntu.edu.sg, **Matri. Number:** G1902089A

ABSTRACT:

This programming assignment presents the evaluations and comparisons of several particle swarm optimization (PSO) algorithms learnt during the graduate-level course EE6227: Genetic Algorithms and Machine Learning. Specifically, four PSOs, i.e., the classic PSO, the comprehensive learning-based PSO (CLPSO) [1], the unified PSO (UPSO) [2], and the fitness-distance-ratio-based PSO (FDR-PSO) [3] are evaluated on eight test functions. With different tuning parameters, the convergence performances of these PSO algorithms are sincerely compared and discussed. What is more, the mean and variance of each algorithm, as well as the convergence curve, is illustrated and compared. Experimental results verify the effectiveness and superiority of the CLPSO and FDR-PSO algorithms for solving multimodal function optimization problems.

KEYWORDS: Particle Swarm Optimization, Multimodal Functions, Convergence.

I. PARTICLE SWARM OPTIMIZATION

Particle swarm optimization (PSO), an evolution method based on computational intelligence, is conceptually simple and easy to implement [4,5]. The stochastic search strategy of PSO stimulates biological activities in nature and exhibit excellent performance in solving global optimization problems of multimodal functions. With parallel search capability, PSO algorithms is more efficient than evolutionary algorithms. The basic procedures (as shown in Figure. 1) of PSO algorithms can be summarized as follows:

1. Initialize all particles and randomly generate the position and velocity particles. Parameter *pbest* of the individual particle is set as the historical optimal value of the best position explored in the current group, and parameter *gbest* of the individual particle is set as the historical optimal value of the best position explored in the all groups.
2. Calculate the fitness value of each particle.

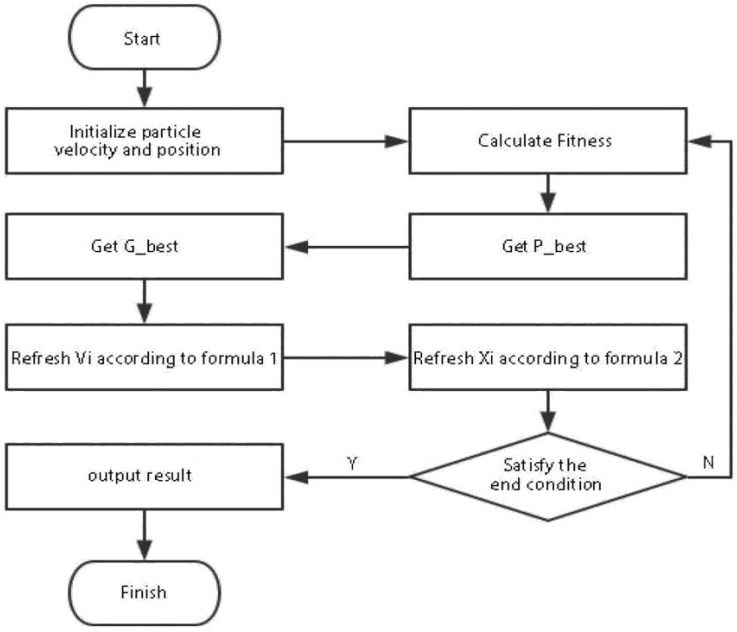


Figure. 1 Basic Procedures of PSO Algorithms

3. For each particle, compare its historical optimal fitness value with the fitness value of the best position found so far in the group. If the current fitness value is better, then select it as the current optimal position.
4. For each particle, compare its historical optimal adaptive value with the updated value of the best position has been experienced. If the current fitness value is better, then select it as the current optimal position.
5. Refresh the d-th dimension of the i-th particle based on procedures as follows:

$$V_i^d \leftarrow V_i^d + c_1 * rand1_i^d * (pbest_i^d - X_i^d) + c_2 * rand2_i^d * (gbest_i^d - X_i^d)$$

$$X_i^d \leftarrow X_i^d + V_i^d$$

where $X_i = (X_i^1, X_i^2, \dots, X_i^D)$ denotes the position of the i-th particle, $V_i = (V_i^1, V_i^2, \dots, V_i^D)$ denotes the velocity of the i-th particle, $pbest_i = (pbest_i^1, pbest_i^2, \dots, pbest_i^D)$ denotes the historical position with the best fitness value found so far for the i-th particle, $gbest_i = (gbest_i^1, gbest_i^2, \dots, gbest_i^D)$ denotes the historical position with the best fitness value found so far for the whole population, c_1 and c_2 denote the acceleration constants which reflect the weight of each particle in terms of pulling toward $pbest$ and $gbest$, respectively, $rand1_i^d$ and $rand2_i^d$ denote randomly generated numbers in range [0,1]. What is more, the velocity of a particle w.r.t. each dimension is clamped to a maximum magnitude denoted by V_{max}. The velocity will be assigned to $sign(|V_i^d|)V_{max}^d$ if $|V_i^d|$ exceed a predefined constant value V_{max}^d.

6. Continue to process if termination condition is not met; otherwise, break the loop. The termination condition can be set as a good adaptive value or reached the predefined iteration number.

To improve the convergence performance of PSO, Shi and Eberhart [4,5] introduced an inertia weight ω in the basic formulation, i.e.,

$$V_i^d \leftarrow \omega \times V_i^d + c_1 * rand1_i^d * (pbest_i^d - X_i^d) + c_2 * rand2_i^d * (gbest_i^d - X_i^d)$$

$$X_i^d \leftarrow X_i^d + V_i^d$$

The inertia weight ω can be used to balance the local and global search abilities of a PSO algorithm. Specifically, a small ω is more appropriate for local search and a large ω is more appropriate for global search. Instead of the original PSO, this assignment, we will take PSO with inertia weight ω as the objective for analysis and discussion.

II. Comprehensive Learning PSO

In [1], Liu *et. al* proposed a comprehensive learning-based PSO (CLPSO), which has the following representation:

$$V_i^d \leftarrow \omega \times V_i^d + c^* * rand_i^d * (pbest_{f_i(d)}^d - X_i^d)$$

where $f_i = [f_i(1), f_i(2), \dots, f_i(D)]$ denotes which particle's $pbest$ are to be followed by the i-th particle. $pbest_{f_i(d)}^d$ denotes the corresponding dimension of any particle's $pbest$ including its own $pbest$. The procedures of CLPSO can be summarized as follows:

1. Firstly, randomly select two particles from the whole population which excludes the particle whose velocity has been updated.
2. Compare the fitness values of these two particles' $pbest$ and choose the better one.
3. Use the winner's $pbest$ as the exemplar to learn from for that dimension. If all exemplars of a particle are its own $pbest$, randomly select one dimension to learn from the corresponding dimension of another particles' $pbest$.
4. All the $pbest$ can generate new positions in the searching space using the information derived from different particles' historical optimal positions.

III. Other PSOs

To compare the convergence performance more thoroughly, in this assignment, two more PSO algorithms are to be used, i.e., the unified PSO (UPSO) [2] and the fitness-distance-ratio-based PSO (FDR-PSO) [3]. Details of these two algorithms are omitted here.

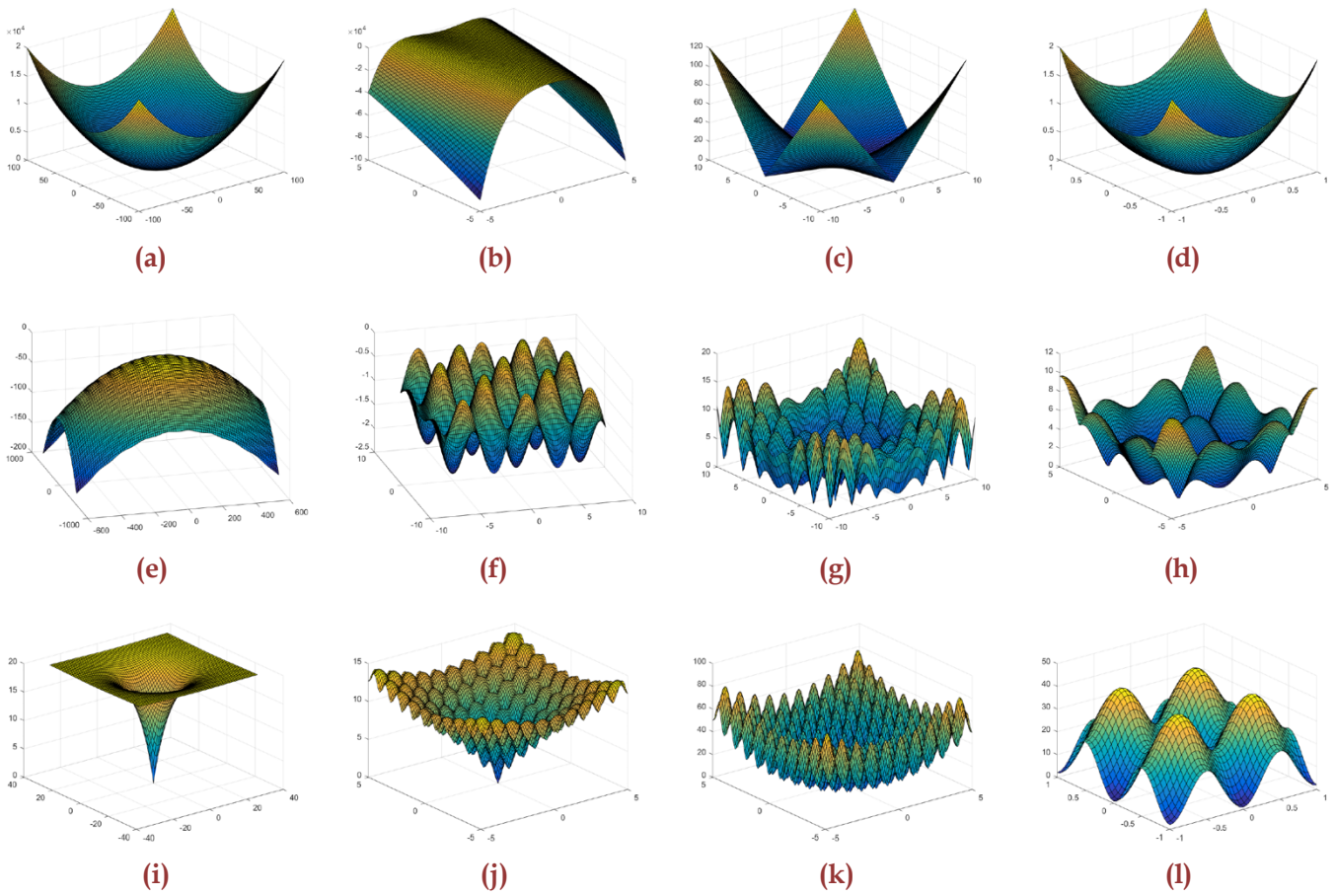


Figure. 2 Some Typical Test Functions.

(a) Sphere in searching space. (b) Rosebrock near extremum. (c) Schwefel P2.22 in searching space. (d) Sum of Different Power in searching space. (e) Griewank in searching space. (f) Griewank near the extremum. (g) Alpine in searching space. (h) Alpine near the extremum. (i) Ackley in searching space. (j) Ackley near the extremum. (k) Rastrigin in searching space. (l) Rastrigin near the extremum.

IV. Test Functions

To evaluate the performance (i.e., convergence speed, error, global search capability, etc.) of a PSO algorithm, some typical benchmark functions are introduced (as shown in Figure. 2). We summarize these test functions as follows:

1. Sphere function $f_1(x)$:

$$f_1(x) = \sum_{i=1}^{D} x_i^2$$

2. Rosenbrock's function $f_2(x)$:

$$f_2(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$$

3. Ackley's function $f_3(x)$:

$$f_3(x) = -20e^{-0.2\sqrt{\frac{\sum_{i=1}^{D} x_i^2}{D}}} - e^{-\sqrt{\frac{\sum_{i=1}^{D} \cos(2\pi x_i)}{D}}} + 20 + e$$

4. Griewanks's function $f_4(x)$:

$$f_4(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

5. Rastrigin's function $f_5(x)$:

$$f_5(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$$

6. Non-continuous Rastrigin's function $f_6(x)$:

$$f_6(x) = \sum_{i=1}^{D} (y_i^2 - 10\cos(2\pi y_i) + 10)$$

$$y_i = \begin{cases} x_i, & |x_i| = \frac{1}{2} \\ \frac{round(2x_i)}{2}, & |x_i| \geq \frac{1}{2} \end{cases}$$

7. Schwefel's function $f_7(x)$:

$$f_7(x) = 418.9829 \times D - \sum_{i=1}^{D} x_i \sin\left(\sqrt{|x_i|}\right)$$

8. Weierstrass function $f_8(x)$:

$$f_8(x) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k\max} \left(0.5^k \cos\left(2\pi \cdot 3^k (x_i + 0.5)\right)\right)\right) - D \sum_{k=0}^{k\max} (0.5^k \cos(2\pi \cdot 3^k \cdot 0.5))$$

What is more, the global optimum, search ranges, and initialization ranges of the above eight test functions are summarized in Table. 1. In Section III, we will use these test functions to evaluate the convergence performance of the PSO algorithms.

Table. I Global optimum, Search Ranges, and Initialization Ranges of Different Test Functions

f	Global Optimum x^*	$f(x^*)$	Search Range	Initialization Range
f_1	[0,0, …,0]	0	$[-100.000, 100.000]^D$	$[-100.000, 50.000]^D$
f_2	[1,1, …,1]	0	$[-2.048, 2.048]^D$	$[-2.048, 2.048]^D$
f_3	[0,0, …,0]	0	$[-32.768, 32.768]^D$	$[-32.768, 16.000]^D$
f_4	[0,0, …,0]	0	$[-600.000, 600.000]^D$	$[-600.000, 200.00]^D$
f_5	[0,0, …,0]	0	$[-0.500, 0.500]^D$	$[-0.500, 0.200]^D$
f_6	[0,0, …,0]	0	$[-5.120, 5.120]^D$	$[-5.120, 2.000]^D$
f_7	[0,0, …,0]	0	$[-5.120, 5.120]^D$	$[-5.120, 2.000]^D$
f_8	[420.96,420.96, …,420.96]	0	$[-500.000, 500.000]^D$	$[-500.000, 500.000]^D$

V. Experiments

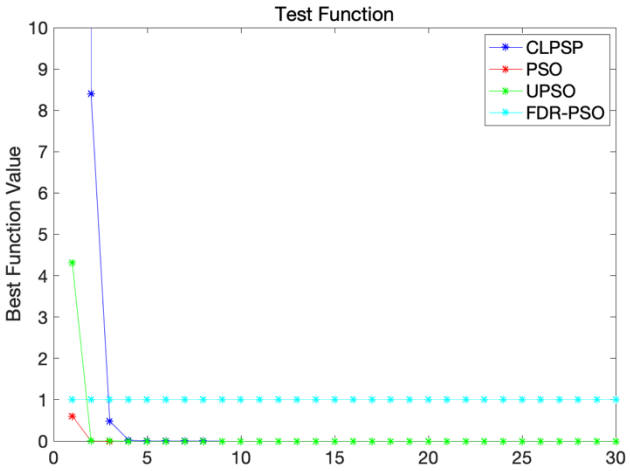
In this section, we illustrate the results of several experiments. Specifically, we will discuss the influence of the tuning parameter ω, the acceleration constant c_1 in classic PSO and c^* in CLPSO. The experiments are tested under ten and therity dimensions.

A. Comparison with Inertia Weight $\omega = 0.5$

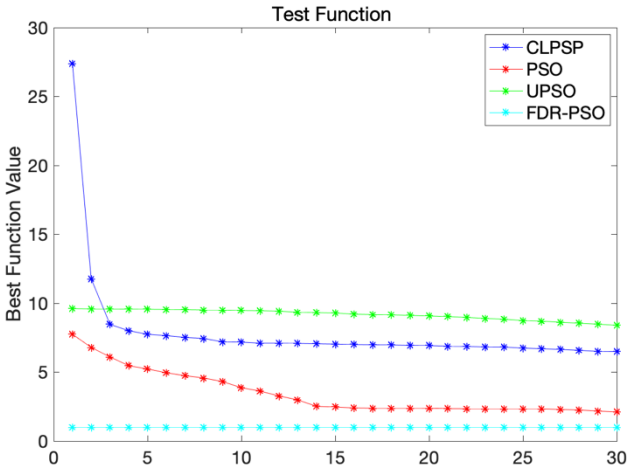
First, let us consider the situation with inertia weight $\omega = 0.5$. To evaluate the convergence performance of the classic PSO, CLPSO, UPSO, and FDR-PSO, all eight test functions are utilized in our experiment and the results are illustrated in Table. II and Figure. 3.

Table. II Experimental Result One

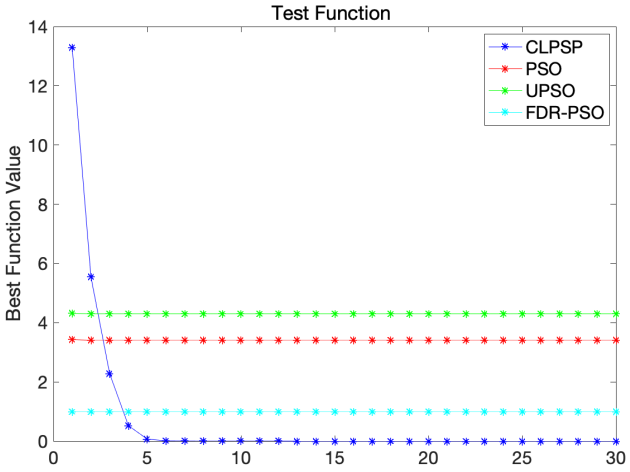
f	$\omega = 0.5$			
	PSO	UPSO	CLPSO	FDR-PSO
f_1	4.82e-106 ± 2.32e-210	3.13e-067 ± 5.47e-133	5.12e-038 ± 1.07e-074	1.56e-164 ± 1.09e-299
f_2	1.59e+000 ± 4.49e-001	4.49e+000 ± 2.56e+000	2.61e+000 ± 4.79e-001	4.02e-001 ± 1.58e-001
f_3	3.57e-001 ± 1.27e-002	4.51e+000 ± 9.51e-001	2.31e-001 ± 2.37e-001	7.43e-001 ± 7.29e-001
f_4	9.43e-002 ± 8.59e-004	2.89e-001 ± 6.65e-002	7.60e-002 ± 6.95e-005	1.18e-001 ± 5.50e-002
f_5	1.61e+001 ± 9.53e+000	1.93e+001 ± 9.73e+001	4.97e-001 ± 2.75e-001	1.29e+001 ± 5.53e+000
f_6	8.10e+000 ± 1.95e+000	1.43e+001 ± 8.93e+000	1.00e+000 ± 1.33e-001	1.90e+001 ± 4.67e+000
f_7	6.25e+002 ± 2.14+e001	1.49e+003 ± 1.24e+002	1.95e+002 ± 6.55+e001	1.79e+003 ± 2.31e+002
f_8	8.05e-001 ± 1.02e-001	2.97e+000 ± 1.79e+000	0 ± 0	1.16e+000 ± 9.46e-001



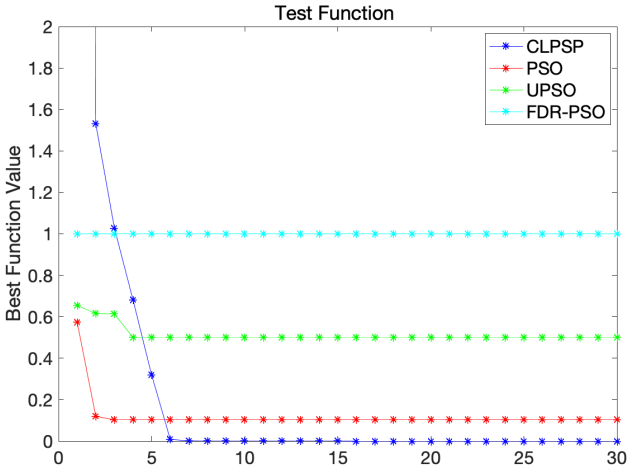
(a) Test Function f_1



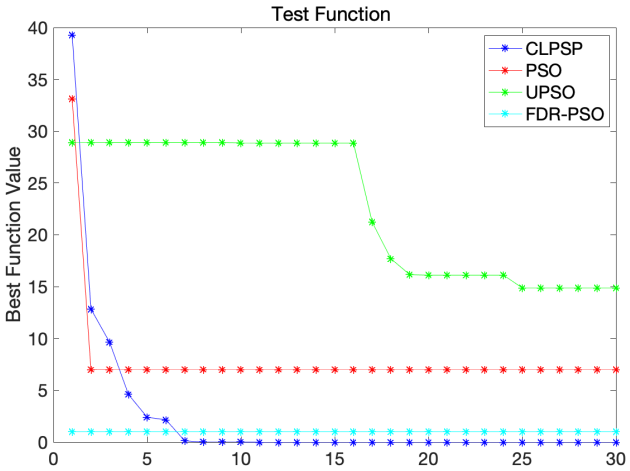
(b) Test Function f_2



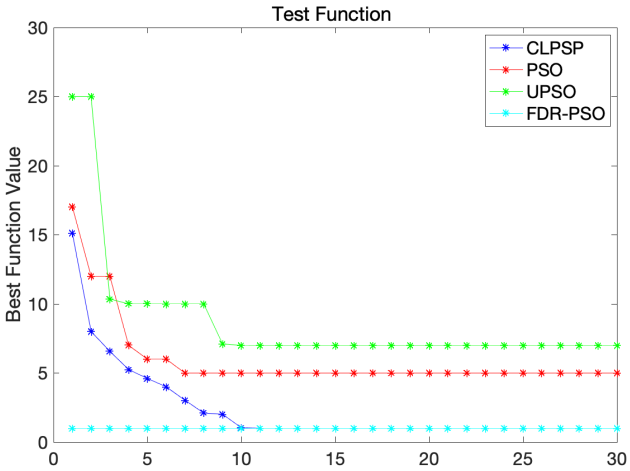
(c) Test Function f_3



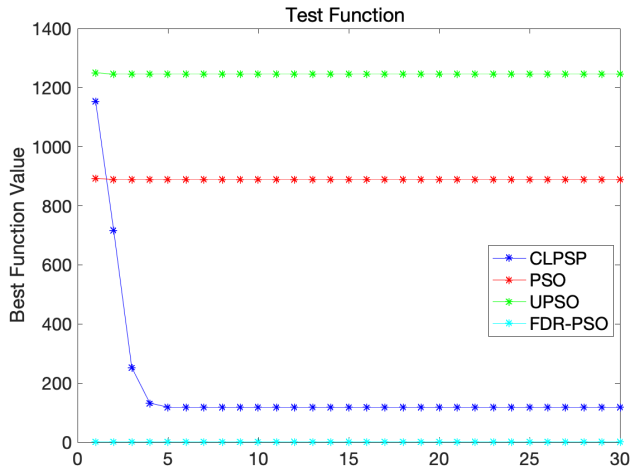
(d) Test Function f_4



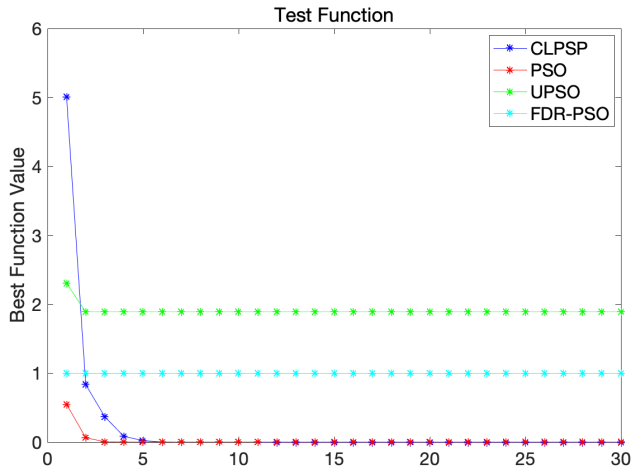
(e) Test Function f_5



(f) Test Function f_6



(g) Test Function f_7



(h) Test Function f_8

Figure. 3 Evaluations of PSOs with $\omega = 0.5$

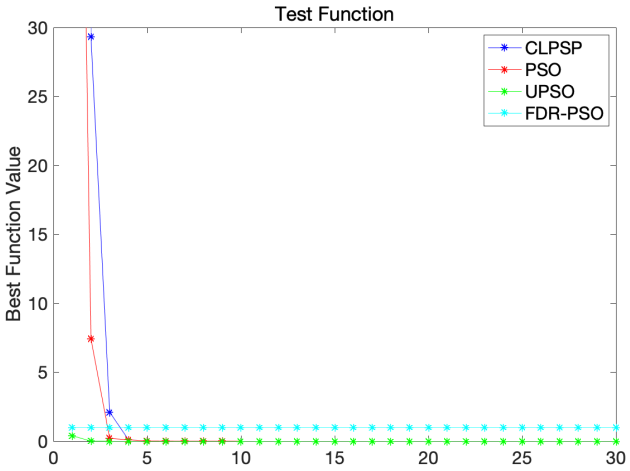
As we can see from Table. II and Figure. 3, the convergence performances of CLPSO and FDR-PSO outperforms the other two algorithms. Specifically, FDR-PSO gives the best performance on the Sphere function $f_1(x)$ and the Rosenbrock's function $f_2(x)$, while CLPSO performs the best on the remaining six test functions.

B. Comparison with Inertia Weight $\omega = 0.7$

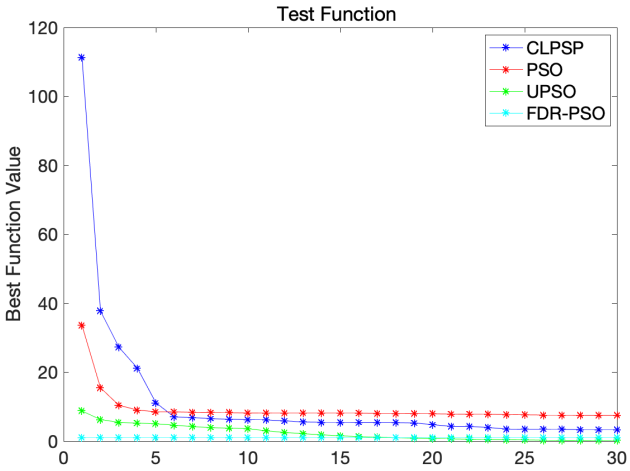
Next, we compare the convergence performance of the aforementioned PSO algorithms under $\omega = 0.7$. The details of our experiment can be seen from Table. III and Figure. 4 as follows.

Table. III Experimental Result Two

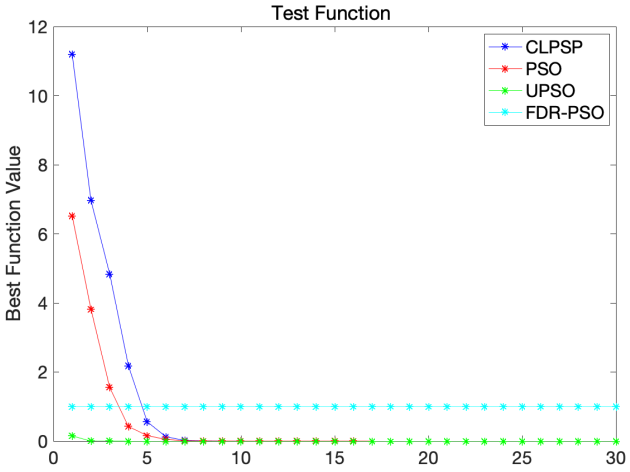
f	$\omega = 0.7$			
	PSO	**UPSO**	**CLPSO**	**FDR-PSO**
f_1	3.55e-022 ± 1.16e-042	4.47e-121 ± 1.07e-240	4.24e-029 ± 1.33e-056	1.74e-072 ± 1.85e-143
f_2	5.51e+000 ± 1.94e+000	1.76e-001 ± 1.18e-002	1.55e+000 ± 1.74e+000	2.44e-001 ± 6.00e-003
f_3	2.49e-012 ± 1.50e-023	1.49e+000 ± 2.37e+000	5.32e-015 ± 3.50e-030	2.57e-001 ± 6.65e-001
f_4	1.17e-001 ± 3.33e-003	8.66e-002 ± 6.90e-002	2.13e-002 ± 1.87e-004	8.81e-002 ± 1.30e-003
f_5	3.08e+000 ± 4.05e+000	1.19e+001 ± 2.49e+001	9.95e-002 ± 9.90e-002	1.52e+001 ± 2.17e+001
f_6	5.09e-001 ± 1.21e+000	8.30e+000 ± 2.57e+000	5.00e-001 ± 5.00e-001	7.60e+000 ± 3.40e+001
f_7	4.38e+002 ± 1.88+e004	1.21e+003 ± 1.48e+005	7.10e+001 ± 1.62+e003	1.50e+003 ± 1.08e+005
f_8	1.09e-013 ± 1.12e-025	9.77e-001 ± 8.20e-001	0 ± 0	8.29e-001 ± 1.27e-000



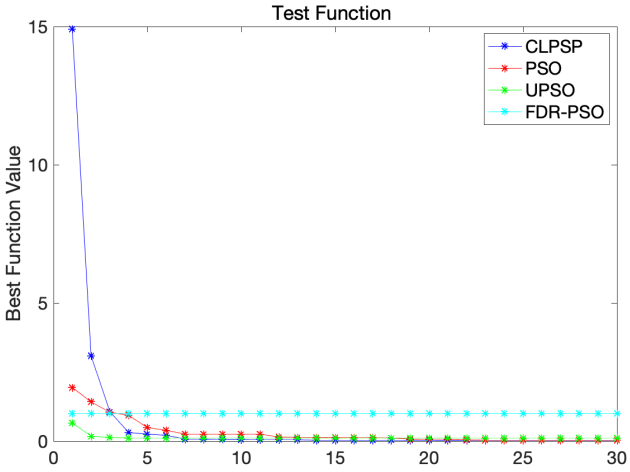
(a) Test Function f_1



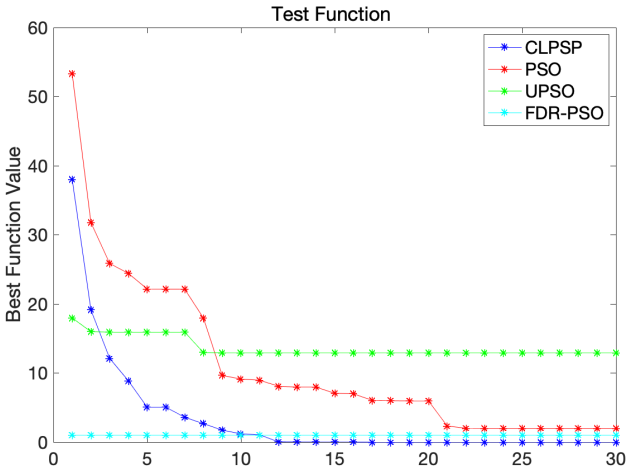
(b) Test Function f_2



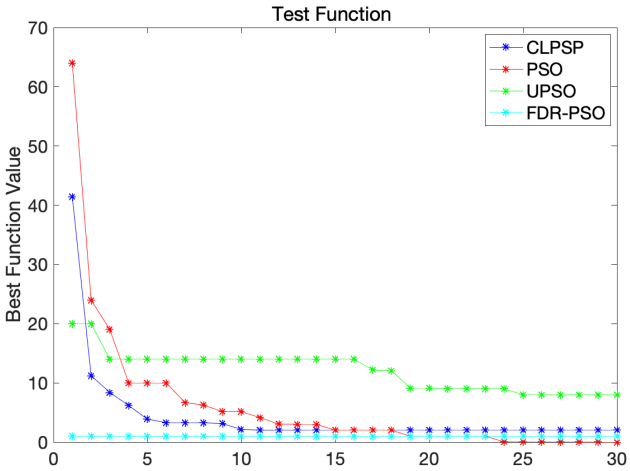
(c) Test Function f_3



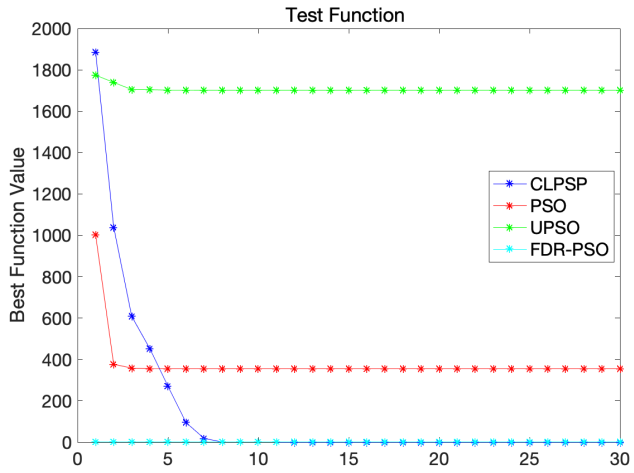
(d) Test Function f_4



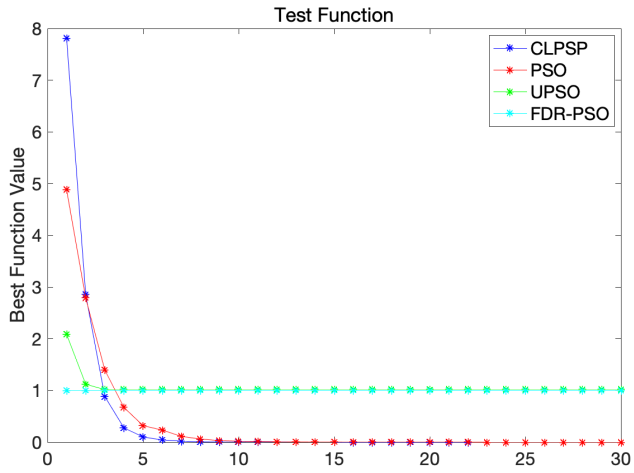
(e) Test Function f_5



(f) Test Function f_6



(g) Test Function f_7



(h) Test Function f_8

Figure. 4 Evaluations of PSOs with $\omega = 0.7$

Obviously, the experimental results are different from the previous one due to the use of different values of the inertia weight ω. Under $\omega = 0.7$, UPSO gives the best performance for the Sphere function $f_1(x)$, FDR-PSO performs the best on the Rosenbrock's function $f_2(x)$. What is more, CLPSO gives the best performance on the remaining test functions.

In general, the commonly used inertia weights ω in PSOs include:

$$\omega_1(k) = \omega_{start} - (\omega_{start} - \omega_{end})(\frac{k}{T_{max}})^2$$

$$\omega_2(k) = \omega_{start} + (\omega_{start} - \omega_{end})[\frac{2k}{T_{max}} - (\frac{k}{T_{max}})^2]$$

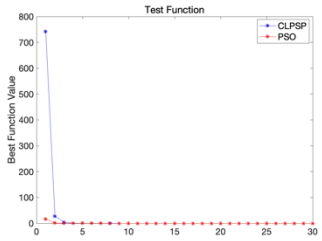
$$\omega_3(k) = \omega_{end}(\frac{\omega_{start}}{\omega_{end}})^{\frac{1}{1+\frac{ck}{T_{max}}}}$$

$$\omega_4(k) = \omega_{max} - k * \frac{\omega_{max} - \omega_{min}}{T_{max}}$$

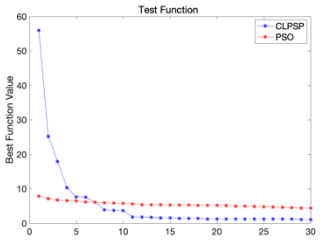
It is worth mentioning that the choose of the inertia weight ω is problem-independent. Generally, a relatively large inertia weight at the beginning of the iteration makes the algorithm maintain a strong global search ability, and a small inertia weight at the end of the iteration makes the algorithm carry out local search more accurately. Therefore, the linear time-varying weight [6] is the most widely used choose in practical applications.

C. Comparison with Different Acceleration Constants

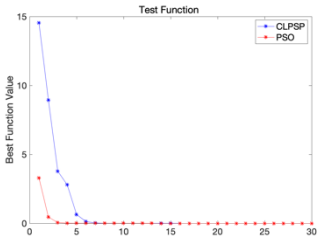
We now evaluate the performance under different acceleration (c_1 in classic PSO and c^* in CLPSO). The experimental results are shown in Figure. 5, Figure. 6, and Table IV.



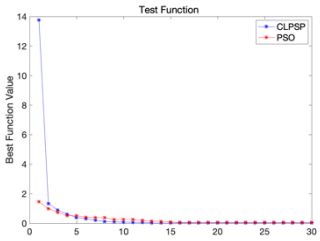
(a) Test Function f_1



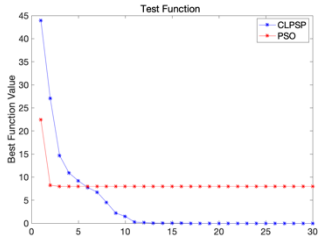
(b) Test Function f_2



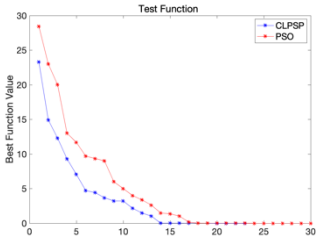
(c) Test Function f_3



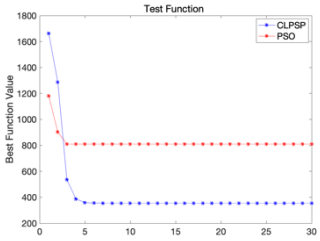
(d) Test Function f_4



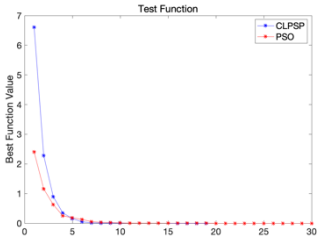
(e) Test Function f_5



(f) Test Function f_6

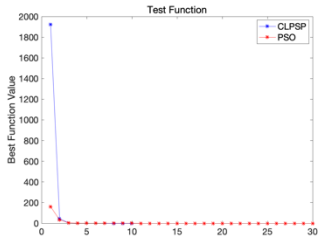


(g) Test Function f_7

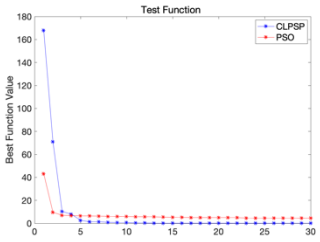


(h) Test Function f_8

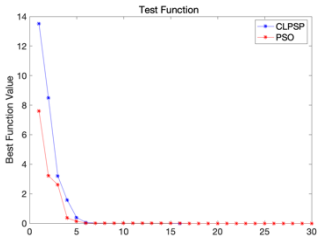
Figure. 5 Evaluations of PSO and CLPSO with $c_1 = c^* = 0.2$



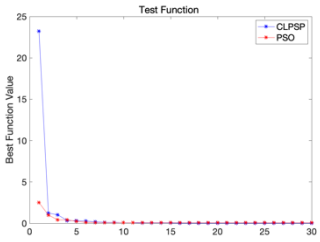
(a) Test Function f_1



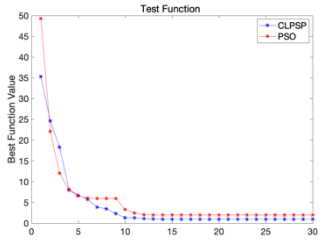
(b) Test Function f_2



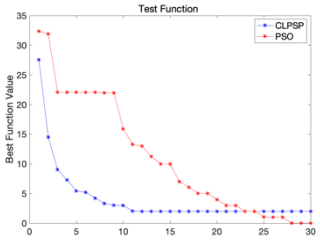
(c) Test Function f_3



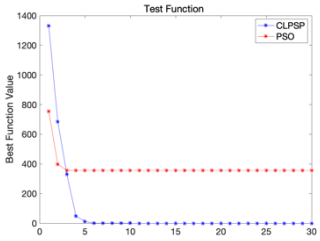
(d) Test Function f_4



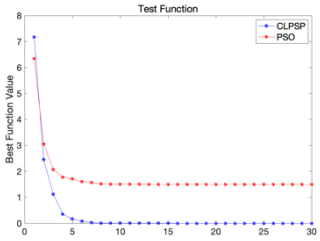
(e) Test Function f_5



(f) Test Function f_6



(g) Test Function f_7



(h) Test Function f_8

Figure. 6 Evaluations of PSO and CLPSO with $c_1 = c^* = 0.8$

Table. IV Experimental Result Three

f	$c_1 = c^* = 0.2$		$c_1 = c^* = 0.8$	
	PSO	CLPSO	PSO	CLPSO
f_1	3.18e-025 ± 1.91e-049	4.27e-030 ± 1.07e-240	9.71e-022 ± 6.01e-042	1.27e-029 ± 2.80e-058
f_2	4.51e+000 ± 4.65e-002	3.02e+000 ± 1.18e-002	3.91e+000 ± 6.39e+000	2.13e+000 ± 2.57e+000
f_3	1.01e-012 ± 3.98e-024	4.26e-015 ± 2.37e+000	3.54e-012 ± 4.47e-023	4.26e-015 ± 2.24e-030
f_4	1.22e-001 ± 2.00e-003	1.33e-002 ± 6.90e-002	8.57e-002 ± 1.50e-003	1.39e-002 ± 1.67e-004
f_5	4.57e+000 ± 6.42e+000	1.62e-011 ± 2.49e+001	3.98e+000 ± 9.46e+000	1.99e-001 ± 1.76e-001
f_6	7.94e-004 ± 5.24e-006	3.00e-001 ± 2.57e+000	1.79e+000 ± 7.75e+000	2.00e-001 ± 4.00e-001
f_7	8.68e+002 ± 5.38+e004	1.53e+002 ± 1.48e+005	4.26e+002 ± 1.93+e004	1.06e+002 ± 1.38e+002
f_8	1.50e-001 ± 2.25e-001	0 ± 0	1.50e-001 ± 2.25e-001	0 ± 0

As we can see from our experimental results, the selection of the acceleration terms is task-oriented. Different setting of the acceleration constants leads to different convergence performance. Specifically, for CLPSO, Rastrigin's function $f_5(x)$ gives better results under 0.2; while for PSO, Non-continuous Rastrigin's function $f_6(x)$ performs better. In practice, the acceleration term can be synchronous time-varying. According to the research of Suganthan et. al [1], these learning factors are set with values in range $[c_{min}, c_{max}]$, and the maximum number of iterations if set as $Iter_{max}$. The acceleration terms are then set as:

$$c_1 = c_2 = c_i = c_{max} - \frac{c_{max} - c_{min}}{Iter_{max}} * i$$

where the learning factors at the i-th iteration is taken as the synchronous linear reduction of two acceleration terms. Due to the time-varying characteristics of these factors, better results can be achieved.

VI. Conclusion

In this assignment, we evaluate the convergence performances of four PSO algorithms learnt during the graduate-level course EE6227: Genetic Algorithms and Machine Learning, i.e., the classic PSO, UPSO, CLPSO, and FDR-PSO. Different settings of the tuning parameters are discussed through computer simulations. Experimental results verify the effectiveness and superiority of CLPSO and FDR-PSO for solving eight commonly tested multimodal optimization functions.

REFERENCES

[1] J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions," in *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp.281-295, 2006.

[2] K. E. Parsopoulos and M. N. Vrahatis, "UPSO: A Unified Particle Swarm Optimization Scheme," in *Lecture Series on Computational Science*, pp. 868-873, 2004.

[3] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-Distance-Ratio-based Particle Swarm Optimization," in *Proc. Swarm Intelligence Symp.*, pp. 174-181, 2003.

[4] R. C. Eberhart and J. Kennedy, "A New Optimizer using Particle Swarm Theory," in *Proc. 6th Int. Symp. Micromachine Human Sci.*, Hagoya, Japan, pp. 39-43, 1995.

[5] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in *Proc. IEEE Int. Conf. Neural Networks*, pp. 1942-1948, 1995.

[6] Y. J. Gong, J. J. Li, Y. Zhou, Y, Li, H. S. Chung, Y. H. Shi, and J. Zhang, "Genetic Learning Particle Swarm Optimization," in *IEEE Trans Cybernetics*, vol. 46, no. 10, pp. 2277-2290, 2017.