Artvin Çoruh Üniversitesi Mühendislik ve Fen Bilimleri Dergisi
Cilt:1 No: 1, 2023 (25-37)

Artvin Coruh University Journal of Engineering and Sciences
Vol: 1 No: 1, 2023 (25-37)

Araştırma Makalesi / Research Paper

On Particle Swarm Optimization Variants for Solution of Some Objective Functions

Hasan BAŞAK[*1], **Kadri DOĞAN**[2]

[1]Artvin Coruh University, Faculty of Engineering, Dept. of Electrical-Electronics Eng. Artvin/TURKEY
[2]Artvin Coruh University, Faculty of Engineering, Dept. of Basic Sciences, Artvin/ TURKEY
hasanbasak@artvin.edu.tr, dogankadri@artvin.edu.tr

Received/Geliş Tarihi: 02.05.2023 **Accepted/Kabul Tarihi:** 15.06.2023

Abstract: Particle Swarm Optimization (PSO) is a widely used metaheuristic algorithm in the field of optimization. Over the years, several variants of PSO have been proposed to improve its performance and overcome its limitations. This study focuses on the comparison of the performance of different PSO variants by solving benchmark functions. We have selected five PSO variants, including constant inertia weight PSO, random inertia weight PSO, time-varying inertia weight PSO, inertia weight-free PSO, nonlinear inertia weight PSO and adaptive inertia weight PSO. These variants have been implemented in MATLAB and tested on some benchmark functions. The results of the experiments show that the performance of the PSO variants changes significantly depending on the benchmark function. However, overall, the adaptive inertia weight PSO variant has shown superior performance compared to the other variants. This variant is capable of finding the global optimum solution with higher accuracy and in a shorter time compared to the other variants.

Keywords: Optimization, PSO, Variants, Benchmark Functions

Bazı Amaç Fonksiyonların Çözümü için Parçacık Sürü Optimizasyon Varyantları Üzerine bir Çalışma

Özet: Parçacık Sürü Optimizasyonu (PSO), optimizasyon alanında yaygın olarak kullanılan metasezgisel bir algoritmadır. Yıllar boyunca, performansını iyileştirmek ve sınırlamalarının üstesinden gelmek için çeşitli PSO varyantları önerilmiştir. Bu çalışma, kıyaslama fonksiyonlarını çözerek farklı PSO varyasyonlarının performansını karşılaştırılmasına odaklanmaktadır. Sabit atalet ağırlıklı PSO, rastgele atalet ağırlıklı PSO, zamanla değişen atalet ağırlıklı PSO, atalet ağırlıksız PSO, doğrusal olmayan atalet ağırlıklı PSO ve uyarlanabilir atalet ağırlıklı PSO dâhil olmak üzere PSO varyantları seçilmiştir. Bu varyasyonlar MATLAB'da gerçekleştirilmiş ve kıyaslama fonksiyonlarında test edilmiştir. Deneylerin sonuçları, PSO varyantlarının performansının kıyaslama fonksiyonuna bağlı olarak önemli ölçüde değiştiğini göstermektedir. Bununla birlikte, genel olarak, uyarlanabilir atalet ağırlıklı PSO varyantı, diğer varyantlara kıyasla üstün performans göstermiştir. Bu varyant, global optimum çözümü diğer varyantlara göre daha yüksek doğrulukta ve daha kısa sürede bulabilmektedir.
Anahtar Kelimeler: Optimizasyon, PSO, Varyasyonlar, Kıyaslama Fonksiyonları

1. Introduction

Optimization is the process of finding the best solution for a given problem within a certain set of constraints. In recent years, optimization algorithms have gained significant attention, especially in the fields of engineering, computer science, finance, and machine learning (Gogna and Tayal 2013). Many realistic optimization issues demand costly computation-based assessments in order to find the optimal solution. The optimization method should be carried out speedily and it should not be overly

*: *Sorumlu Yazar (Corresponding Auhor)*

complicated due to various limits in research such as time requirements and computer resource constraints (Tanweer et al. 2015). A lot of common optimization techniques need a lot of function evaluations. These algorithms often provide acceptable results by utilizing their unique information transmission methods in conjunction with a variety of first-candidate solutions in various fitness evaluations. These procedures often consume significant computational resources and require a substantial amount of time to execute since they evaluate each potential solution. The study and creation of effective optimization algorithms for assessing a small number of functions is therefore a new and expanding research topic. Several novel ideas have been proposed and published recently. These techniques with constrained function evaluations have produced some satisfying results (Rueda and Erlich 2015).

Wilson (2000) first put out the swarm idea in 1975. Each member of a swarm can use the discoveries and experiences of the others to escape from predators and can find food. Each bird in a swarm can identify where it is within the swarm. Every individual will observe neighbouring individuals' flight motions to modify its own flight trajectory, giving the impression that a single entity is in charge of the whole swarm (Zhang 2015; Reynolds 1987) Particle swarm optimization (PSO) is an optimization algorithm that was first proposed by Kennedy and Eberhart (1995). PSO is inspired by the social behaviour of bird flocking or fish schooling. In PSO, a population of particles is used to search for the best solution of a problem. Each particle represents a potential solution and moves through the search space based on its own experience and the experience of the swarm. The position and velocity of each particle are updated based on its own best solution and the best solution found by the swarm PSO has several advantages, such as its simplicity, fast convergence speed, and ability to handle nonlinear and high-dimensional problems. However, PSO has some limitations, such as its sensitivity to parameter tuning, premature convergence, and the lack of a global search strategy (Imran et al. 2013). PSO blends evolutionary calculations with social psychology concepts from socio-cognition agents. It uses a swarm of particles to represent the potential solutions to the objective issue when applied to optimization processes. Each particle will move in the direction of the problem's probable solution after a search has started, based on its own and the partner particles' investigations. The PSO's ease of implementation and the limited number of adjustable parameters are its two main benefits. The inertia weight (w), one of the parameters in Particle Swarm Optimization, plays a crucial role in achieving a balance between exploration and exploitation. In recent years, several variants of PSO have been proposed to enhance its performance and overcome its limitations such as premature convergence and slow convergence speed. This paper investigates some variants of PSO with inertial weights such as constant, random, linear time-varying, nonlinear and adaptive. We compare the performances of PSO variants using different types of benchmark functions.

3. Particle Swarm Optimization

The standard PSO contains the following four items:
1. Determine the objective function.
2. Set parameters.
The basic parameters of the PSO include:
- (i) Space dimension
- (ii) Particle swarm size
- (iii) Location constraint

(iv) Velocity constraint
(v) Number of iterations
(vi) Inertia weight
(vii) Learning factor: The ranges of the independent variables should be considered while determining the learning factor. Particle and particle swarm learning factors are the two different categories of learning factors. Typically, a value between 0 to 5 can be used.

3. Initialize particle swarm
4. Update velocity and location

Updating velocity and position is the essence of the standard PSO. The function velocity and position, which is called the PSO algorithm, is as follows:

$$v^{k+1}(m,n) = wv^{k}(m,n) + r_1 c_1\left(xp^{k}(m,n) - x^{k}(m,n)\right) + r_2 c_2\left(xg^{k}(n) - x^{k}(m,n)\right) \quad (1)$$

$$x^{k+1}(m,n) = x^{k}(m,n) + v^{k+1}(m,n) \quad (2)$$

Where $v^{\mathrm{k}}(m,n)$ is the velocity of the m^{th} particle for the n^{th} dimension at the k^{th} iteration, $x^{\mathrm{k}}(m,n)$ is the current position of the m^{th} particle for the n^{th} dimension at the k^{th} iteration. $xp(m,n)$ represents the position of the best solution that the m^{th} particle has achieved so far, for the n^{th} dimension of the problem. $xg(n)$ is the current global best obtained so far by the particle swarm optimization for the n^{th} dimension of the problem. $w,\ c_1,\ c_2$ and $r_1,\ r_2$ are defined as inertia weight, single particle's learning factor, particle swarm's learning factor and random values in [0,1], respectively (Zhang 2022).

The hybrid optimization algorithm is an effective combination of a metaheuristic optimization algorithm with another optimization algorithm that can exhibit more stable behaviour and greater flexibility against complicated and difficult problems. Local search algorithms use a well-specified neighbourhood mechanism to recursively explore the search space for a better answer than an already existing one. Metaheuristics are made up of iterative processes that successfully integrate many sub-heuristics to find a search space. To locate global optimal areas, certain learning algorithms are employed. Natural approaches known as population-based metaheuristics investigate the search space by manipulating the population, and the outcomes heavily depend on these particular manipulative techniques. Compared to other trajectory approaches, which are easily impacted by local optima, population-based metaheuristics methods are better at characterizing local optima. Because of this, metaheuristic hybrids that effectively combine the advantages of population-based and trajectory approaches are typically quite effective and successful (Blum et al. 2008; Osman and Laporte 1996). During the early search phase, the standard PSO technique often converges quickly before slowing down. It frequently has slow convergence and becomes locked in local minima. Moreover, the inertia weights, $w,\ c_1\ and\ \ c_2$ are important variables affecting the standard PSO convergence. Various inertia weighting strategies in the velocity update equation (1) are reported in Table 1. Where $wmax$ and $wmin$ are maximum and minimum values, $iter_{max}$ is the maximum number of iterations, $iter$ is the current iteration, $PS = \frac{\sum_{i=1}^{M} SC_i}{M}$ where M is the number of particles and SC_i is the successful particle (particle to near solution).

Table 1. Inertia weights.

Label	Inertia Weight Strategy	Adaptation Mechanism/ Reference
W1	$w = c$	Constant (Shi and Eberhart 1998),
W2	$w = 0.5 + \frac{r_1}{2}$	Random (Eberhart and Shi (2001)
W3	$w(iter) = wmin + \frac{iter_{max} - iter}{iter_{max}} * (wmax - wmin)$	Linear time-varying (Shi and Eberhart 1999),
W5	$w(iter) = \left(\frac{2}{iter}\right)^{0.3}$	Nonlinear (Fan and Chiu 2007)
W6	$w(i) = PS * (wmax - wmin) + wmin$	Adaptive (Das et al. 2018; David Reddipogu and Elumalai 2020)

The inertia weight-free PSO has a different particle update strategy from the standard PSO algorithm. The following equations show the calculation of particle positions and velocity in the inertia weight-free PSO algorithm (Jaberipour et al. 2011):

$$v^{k+1}(m,n) = (2r_1 - 0.5)v^k(m,n) + (2r_2 - 0.5)\left(xp^k(m,n) - x^k(m,n)\right) + (2r_3 - 0.5)(xg^k(n) - x^k(m,n)) \quad (3)$$

$$u^{k+1}(m,n) = (2r_4 - 0.5)\left(xg^k(n) - xp^k(m,n)\right) + (2r_5 - 0.5)\left(xg^k(n) - x^k(m,n)\right) \quad (4)$$

$$x^{k+1}(m,n) = xp^k(m,n) + (2r_6 - 0.5)v^{k+1}(m,n) + (2r_7 - 0.5)u^{k+1}(m,n) \quad (5)$$

where r_1, r_2, r_3, r_4, r_5, r_6 and r_7 are defined as random values in [0,1].

Table 2. Benchmark objective functions.

	Objective Function	n	Search Space	$f(x^*)$
F1	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100,100]^n$	0
F2	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10,10]^n$	0
F3	$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100,100]^n$	0
F4	$f_4(x) = \max_i \{\lvert x_i \rvert : 1 \leq i \leq n\}$	30	$[-100,100]^n$	0
F5	$f_4(x) = \sum_{1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	$[-5,5]^n$	0.00030
F6	$f_6(x) = \frac{\pi}{n}\left\{10sin(\pi y_i) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1}) + (y_{n-1})^2]\right\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50,50]^n$	0

4. Results and Discussion

This section gives results of PSO variants to evaluate their performance. We conducted experiments with six different benchmark functions given in Table 2. Figure 1 shows a comparison of PSO variants on objective function F1. The inertia weight-free PSO (WFPSO) obtained the worst performance amongst variants. The random inertia weight PSO (RPSO) converged faster than other variants do but the adaptive inertia weight PSO obtained the lowest value of the objective function with 400 iterations (see Table 3). Convergence curves of PSO variants for the objective function $F2$ are illustrated in Figure 2. It can be seen from Table 4 that the adaptive inertia weight PSO achieved the lowest values with less iteration. For this objective function, the second-best algorithm is the linear time-varying inertia weight PSO (LTVPSO). Convergence curves for the objective function F3 is displayed in Figure 3. The constant inertia weight PSO is the fastest algorithm. However, the adaptive inertia weight PSO obtained the lowest value of the objective function with 500 iteration (Table 5). Similarly, Figures 4, 5 and 6 compare the convergence curves of PSO variants for objective functions 4 , $F5$ and $F6$ respectively. For objective function F4, WFPSO has the worst performance and the adaptive inertia weight PSO achieved the lowest values with 500 iterations (Table 6). For objective function F5, the adaptive inertia weight PSO converged faster than other do and this algorithm obtained the lowest value with 86 iterations (Table 7). For objective function F6, the LTVPSO is the fastest algorithm and the adaptive inertia weight PSO achieved the lowest values with 355 iterations (Table 8). Overall, the results of the experiments show that the performance of the PSO variants changes significantly depending on the benchmark function. However, the adaptive inertia weight PSO variant has shown superior performance compared to the other variants. The adaptive inertia weight PSO has achieved the lowest value in lesser iterations (at the 86^{th} iteration) on objective function F5, see Table 7.

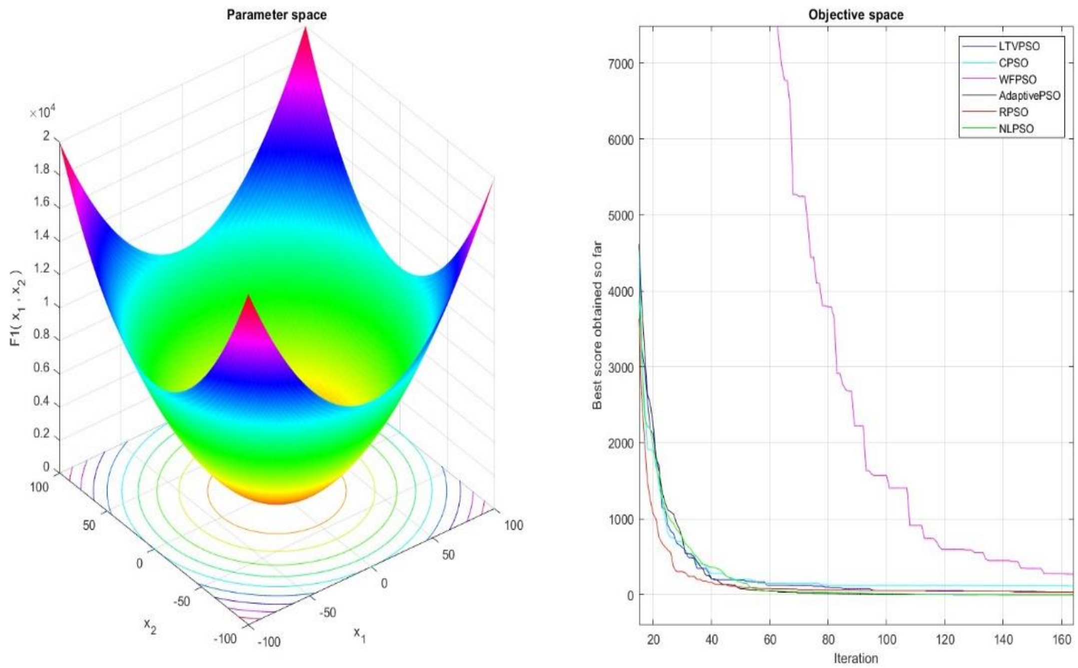

Figure 1. Convergence curves of PSO variants for the objective function $\boldsymbol{F1}$

Table 3. Numerical values of the F1 objective function for comparison of PSO variants

n	LTVPSO Values	CPSO Values	WFPSO Values	Adaptive PSO Values	RPSO Values	NLPSO Values
1	70753,24575	58381,53455	68284,75105	68688,64554	73463,14213	64866,17248
2	63241,43504	58381,53455	68284,75105	66756,80839	67189,94540	58239,03450
3	55221,17496	51186,05980	68284,75105	57577,59383	57276,69296	51991,24550
4	46823,61096	45558,77072	68284,75105	49802,07221	49165,34710	44714,78627
5	40636,88841	40906,27674	68284,75105	42953,55001	42166,04481	40232,61805
6	34737,96513	33712,08192	68284,75105	36505,65862	34505,68711	37717,18519
7	28698,19385	27608,88016	68284,75105	30087,28495	28751,51068	32293,82142
⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	52,81903	119,34873	1353,88381	8,38954	39,40401	5,82270
⋮	⋮	⋮	⋮	⋮	⋮	⋮
200	13,05715	98,33623	198,10067	0,05547	16,85187	0,20752
⋮	⋮	⋮	⋮	⋮	⋮	⋮
300	0,58885	98,33623	129,17871	0,00088	12,21267	0,06621
⋮	⋮	⋮	⋮	⋮	⋮	⋮
400	0,01587	89,58545	98,69304	**0,00001**	8,14288	0,06607
⋮	⋮	⋮	⋮	⋮	⋮	⋮
427	0,00326	89,58545	98,68388	0,00001	7,75599	0,06607
428	0,00271	89,58545	98,68388	0,00000	7,40413	0,06607
⋮	⋮	⋮	⋮	⋮	⋮	⋮
500	0,00007	84,59766	79,05413	0,00000	3,86110	0,06606

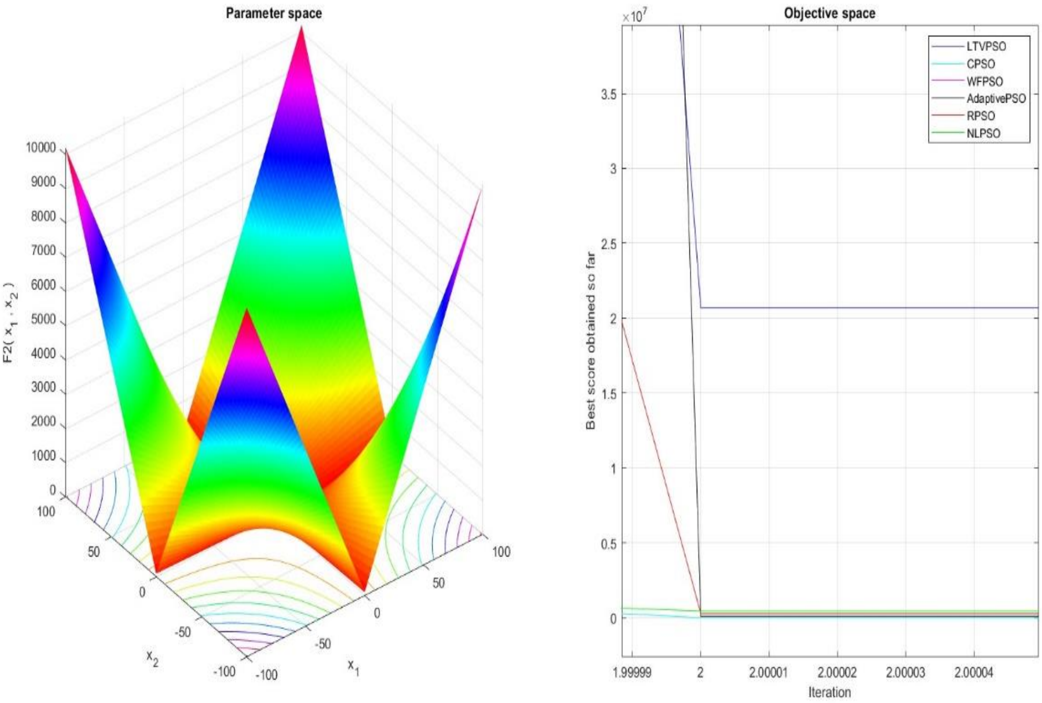

Figure 2. Convergence curves of PSO variants for the objective function $F2$

Table 4. Numerical values of the objective function F2 for comparison of PSO variants

n	LTVPSO Values	CPSO Values	WFPSO Values	Adaptive PSO Values	RPSO Values	NLPSO Values
1	6114142570308,11000	24077928152,86930	345140455978902,00000	15087874576319,00000	1687597775385,29000	16691266620,15370
2	20701549,85748	3643,40407	5777329824441,69000	98788,09879	280523,60820	450262,39038
3	20701549,85748	1438,80043	5777329824441,69000	98788,09879	280523,60820	450262,39038
4	1100,63636	1438,80043	5777329824441,69000	98788,09879	222576,24714	83,32132
5	216,30597	1438,80043	5777329824441,69000	61,18740	115174,39282	83,32132
6	216,30597	1438,80043	5777329824441,69000	50,35392	60453,66763	72,27122
7	216,30597	536,75706	5777329824441,69000	50,35392	1096,75327	72,27122
⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	52,81903	119,34873	1353,88381	8,38954	39,40401	5,82270
⋮	⋮	⋮	⋮	⋮	⋮	⋮
200	18,05983	61,18577	38,66453	0,15826	17,66575	1,17102
⋮	⋮	⋮	⋮	⋮	⋮	⋮
300	2,94593	50,51487	27,40757	0,01574	13,24167	1,12418
⋮	⋮	⋮	⋮	⋮	⋮	⋮
400	0,34005	50,29958	20,08569	0,00155	9,87452	1,12275
⋮	⋮	⋮	⋮	⋮	⋮	⋮
449	0,05582	44,70033	15,32458	0,00037	9,61013	1,12182
450	0,05582	44,70033	15,32458	0,00037	9,61013	1,12182
⋮	⋮	⋮	⋮	⋮	⋮	⋮
500	0,01127	44,07517	12,43246	**0,00009**	9,61011	1,12177

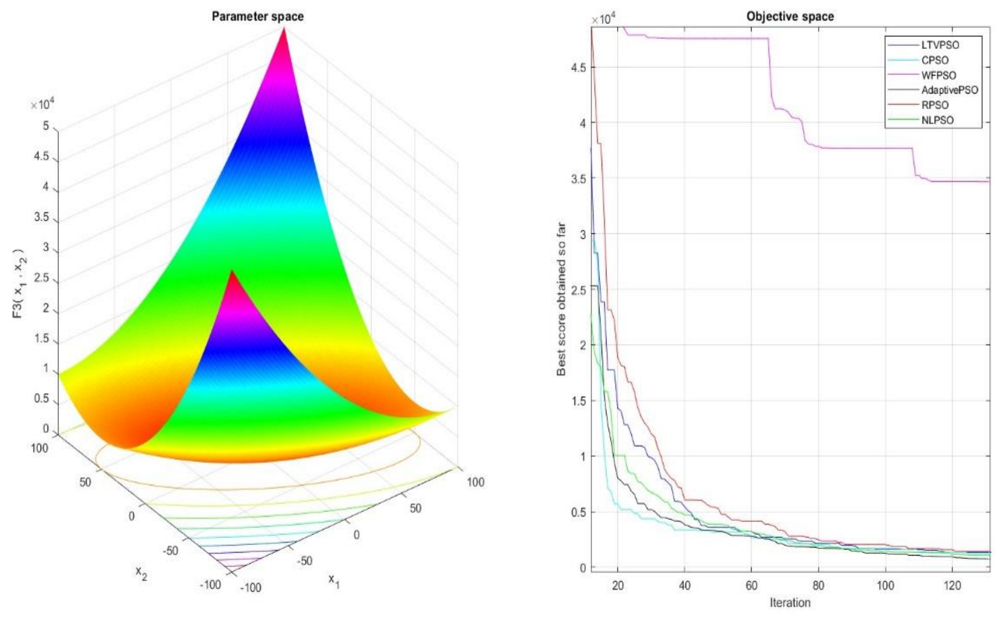

Figure 3. Convergence curves of PSO variants for the objective function $F3$

Table 5. Numerical values of the objective function F3 for comparison of PSO variants

n	LTVPSO Values	CPSO Values	WFPSO Values	Adaptive PSO Values	RPSO Values	NLPSO Values
1	132431,04615	103155,12236	205177,07414	140856,42852	168100,31815	169605,82654
2	132431,04615	103155,12236	205177,07414	140856,42852	168100,31815	148418,25054
3	117347,49849	103155,12236	205177,07414	115689,06299	155330,42532	100300,24614
4	106767,22437	103155,12236	205177,07414	104667,33882	140212,60840	71192,71328
5	87786,95237	82566,03081	205177,07414	103055,32432	116894,81235	61030,66230
6	67057,04190	67765,60821	205177,07414	103055,32432	92009,80862	61030,66230
7	59882,86525	57682,77343	205177,07414	73501,97259	81931,57463	38268,03617
⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	1621,23655	1679,57133	37694,26228	1255,42439	2027,85038	1411,64087
⋮	⋮	⋮	⋮	⋮	⋮	⋮
200	606,04816	957,45778	24744,42578	387,22512	726,09273	693,43846
⋮	⋮	⋮	⋮	⋮	⋮	⋮
300	240,21947	629,51391	16664,27750	136,12930	358,03036	322,77719
⋮	⋮	⋮	⋮	⋮	⋮	⋮
400	116,34421	468,89252	9648,90913	86,31726	306,22395	138,95211
⋮	⋮	⋮	⋮	⋮	⋮	⋮
449	87,32016	404,88214	7358,56321	69,47749	299,00054	91,67042
450	87,30548	404,88214	7358,56160	69,37368	299,00054	91,64130
⋮	⋮	⋮	⋮	⋮	⋮	⋮
500	64,00206	362,68300	7358,55971	56,55063	248,45027	78,06764

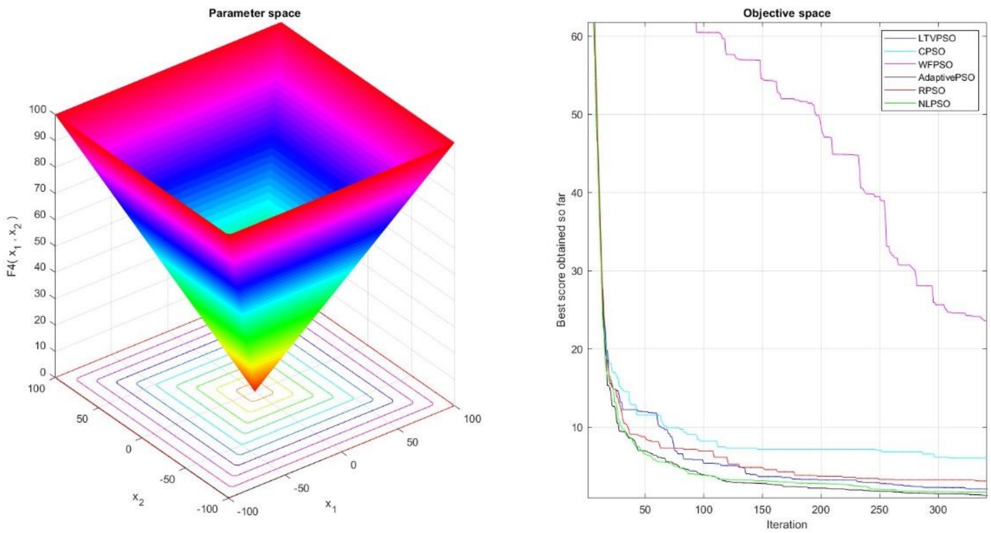

Figure 4. Convergence curves of PSO variants for the objective function $F4$

Table 6. Numerical values of the objective function F4 for comparison of PSO variants

n	LTVPSO Values	CPSO Values	WFPSO Values	Adaptive PSO Values	RPSO Values	NLPSO Values
1	90,03536	86,00834	88,63982	85,42587	88,07936	84,86917
2	84,88253	81,95579	88,63982	85,42587	84,74989	84,86917
3	79,49933	76,55579	88,63982	80,34211	79,80800	81,69993
4	75,06061	71,69579	88,63982	74,98719	73,84699	76,15643
5	69,69421	67,56274	88,63982	69,08211	68,41303	70,34712
6	64,36105	64,83869	88,63982	63,58719	66,09263	64,72991
7	58,93818	60,13253	88,63982	57,98470	61,92738	59,12873
⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	5,39234	8,21923	60,43726	3,96811	6,94893	3,87494
⋮	⋮	⋮	⋮	⋮	⋮	⋮
200	3,25462	7,17209	48,63156	2,21779	3,73279	2,79471
⋮	⋮	⋮	⋮	⋮	⋮	⋮
300	2,29957	6,19587	25,65499	1,46666	3,28332	1,74300
⋮	⋮	⋮	⋮	⋮	⋮	⋮
400	1,74849	4,30861	20,88016	1,11508	3,11876	1,38595
⋮	⋮	⋮	⋮	⋮	⋮	⋮
449	1,64871	4,29268	16,29901	1,00978	2,59656	1,31850
450	1,64871	4,29268	16,29901	1,00900	2,59656	1,31850
⋮	⋮	⋮	⋮	⋮	⋮	⋮
500	1,58286	4,29268	16,08736	**0,93030**	2,37249	1,07871

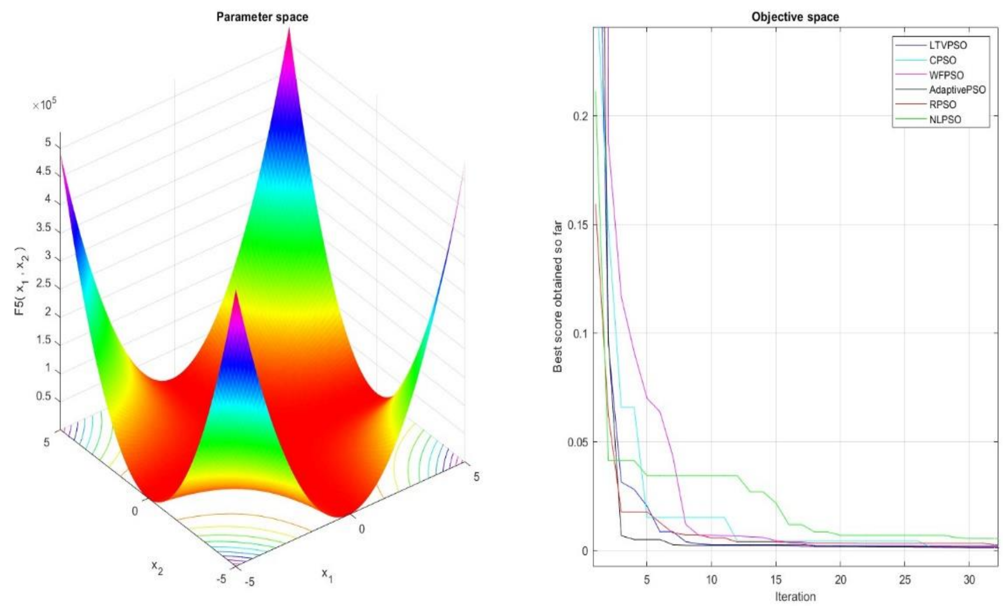

Figure 5. Convergence curves of PSO variants for the objective function $F5$

Table 7. Numerical values of the objective function F5 for comparison of PSO variants

n	LTVPSO Values	CPSO Values	WFPSO Values	Adaptive PSO Values	RPSO Values	NLPSO Values
1	0,39890	0,26904	3,94286	0,55688	127,40654	0,21123
2	0,09505	0,14839	0,18773	0,09659	127,29710	0,04129
⋮	⋮	⋮	⋮	⋮	⋮	⋮
85	0,00109	0,00123	0,00101	0,00100	78,80092	0,00115
86	0,00109	0,00123	0,00101	**0,00099**	78,80092	0,00115
87	0,00109	0,00123	0,00101	0,00099	78,80092	0,00115
88	0,00109	0,00123	0,00101	0,00099	78,56626	0,00115
89	0,00109	0,00123	0,00100	0,00099	78,56626	0,00115
⋮	⋮	⋮	⋮	⋮	⋮	⋮
137	0,00101	0,00123	0,00100	0,00099	62,87574	0,00114
⋮	⋮	⋮	⋮	⋮	⋮	⋮
223	0,00101	0,00119	0,00100	0,00099	62,59744	0,00114
⋮	⋮	⋮	⋮	⋮	⋮	⋮
368	0,00101	0,00119	0,00100	0,00099	49,28112	0,00113
369	0,00101	0,00119	0,00100	0,00099	49,28112	0,00112
⋮	⋮	⋮	⋮	⋮	⋮	⋮
500	0,00101	0,00119	0,00100	0,00099	49,16171	0,00112

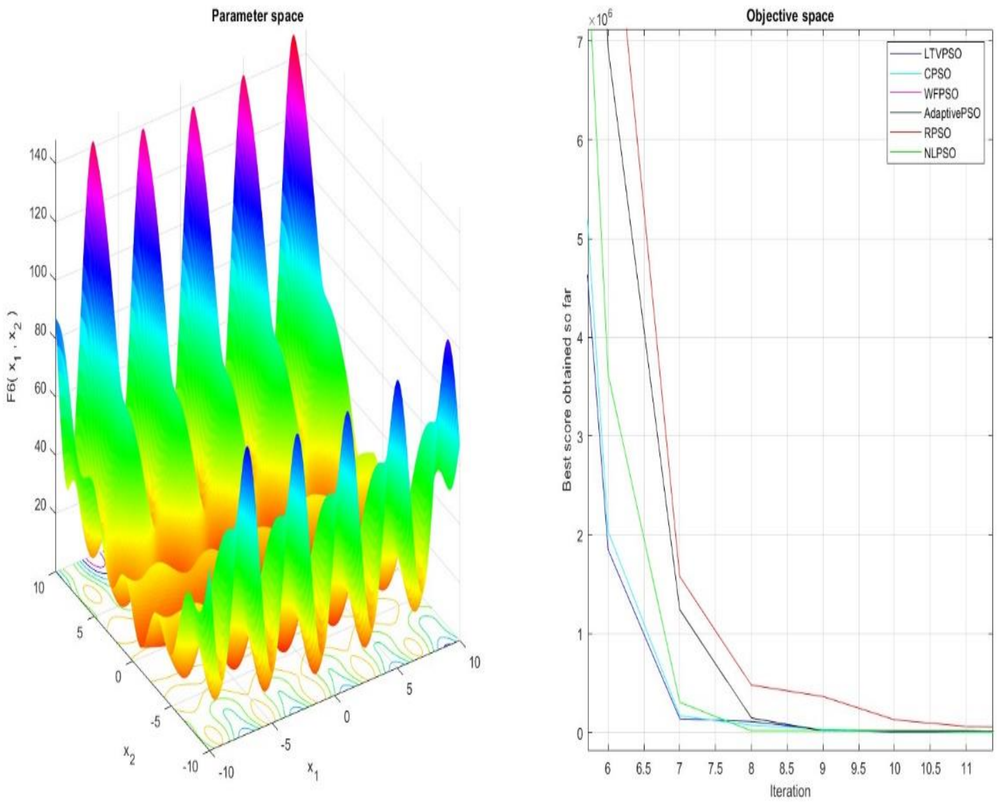

Figure 6. Convergence curves of PSO variants for the objective function $F5$

Table 8. Numerical values of the objective function F6 for comparison of PSO variants

n	LTVPSO Values	CPSO Values	WFPSO Values	Adaptive PSO Values	RPSO Values	NLPSO Values
1	621556339,32325	605104612,10366	357100802,58440	625337349,14362	470844775,35994	644618709,92784
2	307493621,19206	283922035,87197	357100802,58440	357856111,58270	285115044,76162	349985483,07990
3	144908759,18344	116122173,14187	300082305,53473	160396871,77953	172466488,18754	163606962,53006
4	48531788,27206	53690059,40528	203023493,52287	61077600,67139	63422212,05466	69686924,30443
5	11459122,28788	12930495,29380	169810855,25257	22262808,00354	28563612,76297	18499988,06406
6	1848221,77127	2034938,95068	169810855,25257	6920130,46147	9021143,25090	3622395,03469
7	139348,01452	167980,91517	169810855,25257	1243310,89341	1576424,08424	305055,45713
⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	4,36980	16,89716	168650,29260	1,99718	3,61039	1,86408
⋮	⋮	⋮	⋮	⋮	⋮	⋮
200	0,54814	9,41636	642,09455	0,00882	0,69373	0,03194
⋮	⋮	⋮	⋮	⋮	⋮	⋮
300	0,00649	3,86060	21,95242	0,00007	0,26127	0,01676
⋮	⋮	⋮	⋮	⋮	⋮	⋮
354	0,00073	3,86060	13,92241	0,00001	0,15315	0,01676
355	0,00073	3,86060	13,92191	**0,00000**	0,15315	0,01676
⋮	⋮	⋮	⋮	⋮	⋮	⋮
458	0,00001	3,02443	13,00949	0,00000	0,12027	0,01667
459	0,00000	3,02443	13,00949	0,00000	0,12027	0,01667
⋮	⋮	⋮	⋮	⋮	⋮	⋮
500	0,00000	3,02443	9,01510	0,00000	0,12027	0,01667

5. Conclusions

PSO and its variants have been successful in solving various optimization problems. Adaptive inertia weight PSO, in particular, has shown better results than the standard PSO and other variants in the objective function. The adaptive inertia weight PSO adjusts the inertia weight of the algorithm during the search process, based on the performance of the algorithm in previous iterations. This helps to balance the exploration and exploitation of the search space and avoids getting stuck in a local-optima. This adaptation enables the algorithm to explore the search space efficiently and effectively, which results in better optimization results. With its dynamic adjustment strategy, it can adapt to different optimization problems and provide better optimization results compared to other variants. The adaptive inertia weight PSO has achieved the lowest value at the 86^{th} iteration on objective function F5. This study highlights the importance of selecting the appropriate variant of PSO for a specific optimization problem. The results of this study can be used to guide the selection of the most appropriate PSO variant for solving different optimization problems.

Conflict of Interest Statement

There is no conflict of interest between the authors.

References

Blum, C., Roli, A., and Sampels, M. (Eds.). (2008). *Hybrid metaheuristics: an emerging approach to optimization* (Vol. 114). Springer.

Das, R. R., Elumalai, V. K., Subramanian, R. G., and Kumar, K. V. A. (2018). Adaptive predator–prey optimization for tuning of infinite horizon LQR applied to vehicle suspension system. *Applied Soft Computing*, *72*, 518-526.

David Reddipogu, J. S., and Elumalai, V. K. (2020). Hardware in the loop testing of adaptive inertia weight PSO-tuned LQR applied to vehicle suspension control. *Journal of Control Science and Engineering*, *2020*, 1-16.

Eberhart, R. C., and Shi, Y. (2001, May). Tracking and optimizing dynamic systems with particle swarms. In *Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546)* (Vol. 1, pp. 94-100). IEEE.

Fan, S. K. S., and Chiu, Y. Y. (2007). A decreasing inertia weight particle swarm optimizer. *Engineering Optimization*, *39*(2), 203-228.

Gogna, A., and Tayal, A. (2013). Metaheuristics: review and application. *Journal of Experimental & Theoretical Artificial Intelligence*, *25*(4), 503-526.

Imran, M., Hashim, R., and Abd Khalid, N. E. (2013). An overview of particle swarm optimization variants. *Procedia Engineering*, *53*, 491-496.

Jaberipour, M., Khorram, E., and Karimi, B. (2011). Particle swarm algorithm for solving systems of nonlinear equations. *Computers & Mathematics with Applications*, *62*(2), 566-576.

Kennedy, J., and Eberhart, R. (1995, November). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942-1948). IEEE.

Osman, I. H., and Laporte, G. (1996). Metaheuristics: A bibliography. *Annals of Operations research*, *63*, 511-623.

Reynolds, C. W. (1987, August). Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (pp. 25-34).

Rueda, J. L., and Erlich, I. (2015, May). MVMO for bound constrained single-objective computationally expensive numerical optimization. In *2015 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1011-1017). IEEE.

Shi, Y., and Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *Evolutionary Programming VII: 7th International Conference, EP98 San Diego, California, USA, March 25–27, 1998 Proceedings 7* (pp. 591-600). Springer Berlin Heidelberg.

Shi, Y., and Eberhart, R. C. (1999, July). Empirical study of particle swarm optimization. In *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)* (Vol. 3, pp. 1945-1950). IEEE.

Tanweer, M. R., Suresh, S., and Sundararajan, N. (2015, May). Improved SRPSO algorithm for solving CEC 2015 computationally expensive numerical optimization problems. In *2015 IEEE congress on evolutionary computation (CEC)* (pp. 1943-1949). IEEE.

Wilson, E. O. (2000). *Sociobiology: The new synthesis*. Harvard University Press.

Zhang, W. (2015). *Selforganizology: The Science of Self-Organization*. World Scientific.

Zhang, W. (2022). Particle swarm optimization: A Matlab algorithm. *Selforganizology*, *9*(3-4), 35-41.