Evolutionary Intelligence (2021) 14:305–313
https://doi.org/10.1007/s12065-018-0188-7

SPECIAL ISSUE

Particle swarm optimization with adaptive inertia weight based on cumulative binomial probability

Ankit Agrawal[1] · **Sarsij Tripathi**[1]

Received: 22 May 2018 / Revised: 20 October 2018 / Accepted: 31 October 2018 / Published online: 13 November 2018

Abstract
Particle swarm optimization (PSO) is a population oriented heuristic numerical optimization algorithm, influenced by the combined behavior of some birds. Since its introduction in 1995, a large number of variants of PSO algorithm have been introduced that improves its performance. The performance of the algorithm mostly rely upon inertia weight and optimal parameter setting. Inertia weight brings equivalence among exploitation and exploration while searching optimal solution within the search region. This paper presents a new improved version of PSO that uses adaptive inertia weight technique which is based on cumulative binomial probability (CBPPSO). The proposed approach along with four other PSO variants are tested over a set of ten well-known optimization test problems. The result confirms that the performance of proposed algorithm (CBPPSO) is better than other PSO variants in most of the cases. Also, the proposed algorithm has been evaluated on three real-world engineering problems and the results obtained are promising.

Keywords Inertia weight · Particle swarm optimization (PSO) · Exploration and exploitation · Convergence

1 Introduction

The particle swarm optimization is a population oriented meta-heuristic optimization technique which was first proposed by Russell Eberhart and James Kennedy [1]. This algorithm is motivated from the social behavior of some animal groups like fish schools or bird flocks. Similar to other meta-heuristic optimization algorithms, in PSO, a swarm of possible solutions is derived in succeeding iterations. PSO is relatively easy to understand and implement since there are very few parameter settings that are required to tune in comparison to other optimization strategies.

In PSO, each particle represents a prospective solution to an optimization problem. The group of particles (or swarm) fly within search space by trailing the current optimum solutions. Every particle remembers its best coordinate position in the search region which is related to the best position it has attained so far. This position is called pbest. Similarly, the best current position of the particle within whole swarm, called as gbest is also remembered. In PSO, each particle changes its velocity towards its gbest and pbest location after each time step. The basic PSO [1] is slow in most cases and prematurely converges to local optima. The solution to their problem is use of inertia weight. The inertia weight [2] is the primary parameter of the PSO, and has an important part in balancing the exploration and exploitation. After the introduction of inertia weight, many versions of the PSO algorithm have been presented aiming balanced exploration–exploitation trade-off.

Shi and Eberhart [2] examined different constant values of inertia weight and reached to the conclusion that within specific range, PSO search the optimum global solution within a acceptable number of iterations. Mostly with lower value of inertia weight, PSO converges in local optima region and with higher value of inertia weight; it diverges and hence performs little exploration in the search region. For finding an optimum solution in a dynamic environment, a random inertia weight strategy is used by Russell Eberhart and Shi [3] and varies in the range [0.5, 1]. This approach is used to alter exploitation and exploration randomly.

A large number of inertia weight variants use time-varying inertia weight method which utilizes iteration number to determine the value of inertia weight. Most famous among them is linear decreasing inertia weight technique [4], in

✉ Ankit Agrawal
aagrawal.phd2017.cse@nitrr.ac.in

[1] Department of Computer Science and Engineering, National Institute of Technology Raipur, G.E Road, Raipur, Chhattisgarh 492001, India

which the inertia weight linearly drops from ω_{max} to ω_{min}. It is found empirically that in most of the cases inertia weight in range [0.4, 0.9] provides the optimum result. Some other time-varying techniques are simulated annealing [5], sigmoid [6], exponential decreasing [7], and logarithmic decreasing technique [8]. Nickabadi and Ebadzadeh [9] suggests adaptive inertia weight law which controls population diversity and enhances its searching ability. Lei et al. [10] suggests Sugeno function as inertia weight method to auto-tune the global and local search ability to avoid premature convergence problem.

Zhan et al. [11] proposed an evolutionary state estimation technique which uses fuzzy classification method to determine the inertia weight. A Gaussian perturbation-based elitist learning strategy has also been adopted which facilitates the particles to move out of local optima region. As a feedback parameter, Nickabadi et al. [12] utilized a percentage of the particles that have shown improvement when compared to the last iteration. They applied this strategy to solve a real-world engineering problem efficiently. Kessentini and Barchiesi [13] presented an adaptive inertia method which uses the standard deviation of dimensions of particles. This technique improves the exploration and exploitation balance in comparison to other algorithms, which helps in converging to optima.

Bansal et al. [14] reviewed 15 famous variants of PSO and compares their performance. Jordehi et al. [15] and Wang et al. [16] examined all popular PSO variants after applying them to deal with discrete optimization problem. Ghamisi et al. [17] applied hybrid PSO and Genetic Algorithm to deal with pipe and road problem. Similar hybrid approach is utilized to find the compressive strength of rock using PSO based Artificial Neural Network by Momeni et al. [18]. Gao et al. [19] suggested novel strategy so that some particles that are better informed could lead to remaining particles. An improved version of PSO is applied for data classification [20]. Various modified particle swam optimization along with their applications is discussed by Tian et al. [21].

The remaining paper is structured as follows: the background of PSO is overviewed in next section. The proposed inertia weight law is presented in further section. Further paper contains the set of optimization test problems for evaluation of the proposed technique, similar other techniques which are compared with proposed technique, simulation setting and results. It also contains the real-world engineering optimization problems used to evaluate and compare the proposed method with other methods. Finally last section contains conclusion of the paper.

2 Background

The PSO is basically a cooperative approach, in which N number of particles flies through the n-dimensional problem search space. The particle's position depends on its own and its neighbor's past best positions. The ith particle's position is written as: $\overrightarrow{x_i} = \left(x_{i1}, x_{i2}, x_{i3}, \ldots, x_{in}\right)$ which is a n dimensional vector. Similarly, velocity of each particle is also a n dimensional vector represented by a vector is given by: $\overrightarrow{v_i} = \left(v_{i1}, v_{i2}, v_{i3}, \ldots, v_{in}\right)$. At each iteration t, the particles of swarm adjust their position and velocity according to following equations [12, 14]:

$$v_{id}(t+1) = v_{id}(t) + c_1 * R_{1id} * (pbest_{id} - x_{id}(t)) + c_2 * R_{2id} * (gbest_d - x_{id}(t)) \quad (1)$$

$$x_{id}(t+1) = \omega^* x_{id}(t) + v_{id}(t+1) \quad (2)$$

where ω is the inertia weight, R_{1id} and R_{2id} are uniform random positive numbers in range (0,1), $d = 1, 2, \ldots, n$ represents dimensions, c_1 is cognitive and c_2 is social learning parameter respectively, $pbest_{id}$ is ith particle's best previous position, $gbest_d$ is globally best position among all particles of the swarm. It becomes social-only model when $c_1 = 0$ (and $c_2 \neq 0$), and cognition-only model when $c_2 = 0$ (and $c_1 \neq 0$). For success of an optimization algorithm, the tuning between the local and global search is important during a run. In most of the cases, the higher value of inertia weight enhances the ability to explore the region of search space, and the lower value of inertia weight improves the capacity to concentrate the search in the vicinity of the promising region to improve the prospective solution. Therefore, for maintaining equilibrium between exploration–exploitation tradeoff effectively, we introduce a new inertia weight law in this paper which is adaptive in nature. During the algorithm process, the inertia weight is adjusted dynamically using feedback on particle's best positions to alternate exploration and exploitation.

3 Proposed inertia weight strategy

The performance and the capability of the swarm based optimization algorithm depend mainly on the exploration–exploitation tradeoff. The proposed inertia weight strategy i.e. CBPPSO applies the idea that in order to balance the local and global search, particles whose position is not improved, must move towards the particles whose position is improved with respect global optimum. Thus movement of each particle will depend on the adaptive inertia weight that uses cumulative binomial probability of finding the global optimum by particles with improved position.

For calculating inertia weight using this method, at each iteration status of the population is determined. The indicator function that denotes the improvement in position of particle i at time t is given as [12, 22]:

$$I(i,t) = \begin{cases} 0 \; if \quad fit(pbest_i^t) \geq fit(pbest_i^{t-1}) \\ 1 \; if \quad fit(pbest_i^t) < fit(pbest_i^{t-1}) \end{cases}. \quad (3)$$

Further calculate the total particles with improved position at iteration t as:

$$X(i,t) = \sum_{i=1}^{N} I(i,t). \tag{4}$$

Considering each iteration as an experiment, in which each individual particle represents a trial which can have one of either state i.e. in a failure or success state. So the probability of success p and failure q is 0.5 for each particle and particles are independent as the trials are independent of each other. Population size is constant (N) in all the iterations. Here $X(i,t)$ which denotes the total particles with improved position is binomial random variable [22]. The cumulative binomial probability is the probability of getting $X(i,t)$ or fewer particles that have improved their position and is given by the following equation:

$$P(k \le X, N, p) = \sum_{k=0}^{X} \binom{N}{k} p^k q^{N-k} \tag{6}$$

where $X = X(i,t)$. The inertia weight at time t is the function of $P(X)$ and represented as:

$$\omega(t) = f(P(k \le X, N, p)). \tag{7}$$

In our algorithm, the linear function used to map the inertia weight and $P(X)$ is given as:

$$\omega = \omega_{\min} + (\omega_{\max} - \omega_{\min}) * P(k \le X, N, p). \tag{8}$$

The velocity of particle plays an important role as it must not be too low so that the particles cannot jump out of local optima regions and neither is too high so that they could avoid the region having the global optimum. Initially when more number of particles improves their respective position while exploring the global optimum, the cumulative binomial probability of finding the global optimum by the particles that improve their position should be high and later on it decreases while converging towards the global optimum. The Fig. 1 shows above property of the proposed adaptive inertia weight when tested on sphere function. The inertia weight varies in range [0.4, 0.9] and is dynamic in nature. For avoiding the premature convergence the lower bound is set to 0.4. This strategy enhances the capability of balancing the local and global search of the algorithm.

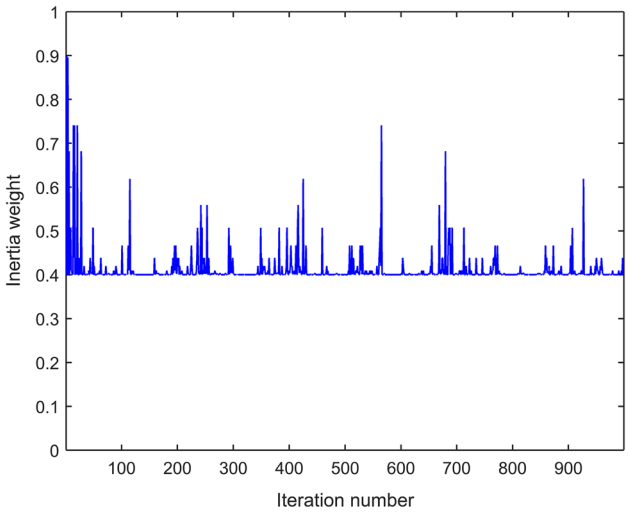


Fig. 1 Inertia weight adaptation in CBPPSO when applied to sphere test function

4 Experimental setup

4.1 Optimization test problems

To measure the performance of the introduced PSO variant, we have used ten well-known benchmark functions [23–29]. The CBPPSO has been compared with four other significant PSO algorithm which are based on different inertia weight methods over these benchmark functions. In Table 1 the benchmark function are listed in which D is number of dimensions, taken as 30.

Functions f_1, f_3, f_4, f_6 are bowl shaped functions. These functions have only global minimum. These functions are convex, continuous and unimodal in nature. Functions f_2, f_5 are Valley shaped functions and mostly used to test gradient optimization algorithms. These functions are also unimodal in nature. Functions f_7, f_8,f_9 are functions with multiple local minima. The outer region of function f_7 is nearly flat and it have a large pit at centre. Functions f_8 and f_9 have multiple regularly distributed local minima. These functions are highly multimodal and the optimization algorithms are most likely to be stuck in one of multiple local minima. All the problems are minimization problems. Best solution and Best fitness for the problem is expressed by x^* and $f(x^*)$ respectively.

4.2 Inertia weight methods used for comparison

In this paper for comparison with proposed approach, we have carefully selected the PSO algorithms that studied the impact of inertia weight strategy. Based on previous comparative performance study [12, 14], the introduced algorithm is simulated and compared with following time varying and adaptive algorithms:

A. *GPSO* [4]:

$$\omega = 0.9 - 0.5\left(\frac{t}{T}\right) \in [0.4, 0.5]$$

Table 1 Optimization test problems used in experiments

Function Name	Test function	Search space	x^*	$f(x^*)$
Sphere function	$f_1 = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$	(0,…,0)	0
Rosenbrock function	$f_2 = \sum_{i=1}^{D-1} [100(x_{i+1}^2 - x_i^2) + (x_i - 1)^2]$	$[-5, 10]$	(1,…,1)	0
Rotated hyper-ellipsoid function	$f_3 = \sum_{i=1}^{D} \sum_{j=1}^{i} x_j^2$	$[-100, 100]$	(0,…,0)	0
Sum squares function	$f_4 = \sum_{i=1}^{D} i x_i^2$	$[-10, 10]$	(0,…,0)	0
Dixon price function	$f_5 = (x_1 - 1)^2 + \sum_{i=2}^{D} i(2x_i^2 - x_{i-1})^2$	$[-10, 10]$	at $x_i = 2^{-\frac{2^i-2}{2^i}}$, for $i = 1, ..., D$	0
Sum of different powers function	$f_6 = \sum_{i=1}^{D} \left\lvert x_i^{i+1} \right\rvert$	$[-1, 1]$	(0,…,0)	0
Ackley function	$f_7 = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}\cos(x_i^2)}\right) - \exp\left(-\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + \exp(1)$	$[-32, 32]$	(0,…,0)	0
Rastrigin function	$f_8 = 10D + \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i)]$	$[-5.12, 5.12]$	(0,…,0)	0
Griewank function	$f_9 = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	(0,…,0)	0
Powell function	$f_{10} = \sum_{i=1}^{D/4} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4]$	$[-4, 5]$	(0,…,0)	0

where t is current step number and T is maximum step number.

B. *Sugeno* [10]:

$$\omega = 0.4 + 0.5\frac{1 - \frac{t}{T}}{1 + s\left(\frac{t}{T}\right)} \in [0.4, 0.5]$$

where s is constant and is taken as 10.

C. *AIPWSO* [12]:

$$\omega = \left(\frac{S(t)}{N}\right) \in [0, 1]$$

where $S(t)$ is total particles with improved best positions and N is the swarm size.

D. *w-PSO* [13]:

$$\omega = 0.9 - 0.4\left(\frac{d(k)}{\max_{1 \le k \le K}\{d(k)\}}\right) \in [0.4, 0.9]$$

where K is a constant (taken as 1000) and d(k) is adaptive vector of K elements.

4.3 Evaluation and comparison criteria

To the ten well-known benchmark functions listed in Table 1, four inertia weight methods along with CBPPSO is tested and their results is compared. All the parameters are kept constant for fair comparison of different inertia weight strategies during the whole experiment so that we can analyze and study the impact of inertia weight in the algorithm. The swarm size is 20 and the dimension is 30 [12]. According to the law presented by Martinez and Gonzalo [30] i.e., $c_1 + c_2 < 4(1 + \omega)$, we took $c_1 = c_2 = 2$. The inertia weight varies in range [$\omega_{\min}$, $\omega_{\max}$] where $\omega_{\min} = 0.4$ and $\omega_{\max} = 0.9$ [4]. The maximum number of functional evaluations (FEs) that each algorithm is allowed to run is fixed to 200,000 for proper comparison among all algorithms. The average results are recorded after running each experiment 30 times.

5 Results and discussion

The best fitness average and standard deviation is recorded in 30 independent runs are listed in Table 2 for each algorithm. The outcomes are compared on the basis of the convergence speed and the final accuracy [13].

- For function f_1 and f_6, CBPPSO yields best solution and it is followed by AIWPSO. The convergence curve is almost straight line indicating that it is approaching towards the optimal solution with constant convergence rate.

Table 2 The mean and the standard deviation of best fitness of five different inertia weight approach on ten benchmark problems in 200,000 function evaluations in 30 runs

Fun	Best fitness mean (best fitness standard deviation)				
	CBPPSO	GPSO	Sugeno	AIWPSO	w-PSO
f_1	7.1632e−137(3.8683e−136)	1.6615e−49(9.0212e−49)	4.9229e−82(2.6511e−81)	6.5670e−114(2.6700e−113)	1.3333e+03(3.4574e+03)
f_2	7.7675e+03(1.9599e+04)	8.6408e+04(7.0973e+04)	6.9299e+04(4.9348e+04)	2.5990e+04(2.8164e+04)	3.2866e+04(5.0152e+04)
f_3	5.1539e+03(9.3962e+03)	1.9184e+04(2.1422e+04)	1.6607e+04(1.8661e+04)	6.5856e+03(1.1922e+04)	5.0108e+03(1.0280e+04)
f_4	1.8333e+02(2.6403e+02)	4.3000e+02(4.8148e+02)	5.2333e+02(4.7827e+02)	1.5333e+02(2.5014e+02)	2.8000e+02(3.8452e+02)
f_5	3.9396e+03(2.1222e+04)	2.5073e+04(5.7882e+04)	1.2283e+04(4.1687e+04)	8.7618e+03(2.7366e+04)	4.0217e+03(2.1622e+04)
f_6	2.3152e−257(0.0000e+00)	8.4450e−93(3.8650e−92)	1.8767e−200(0.0000e+00)	3.1842e−212(0.0000e+00)	5.9589e−38(2.9703e−37)
f_7	1.4091e+00(1.2427e+00)	4.7725e−01(2.6140e+00)	4.7454e−01(2.5991e+00)	1.6503e+00(3.5923e+00)	4.9727e−01(2.6229e+00)
f_8	8.3739e+01(2.8673e+01)	7.2624e+01(2.5950e+01)	9.2580e+01(2.6948e+01)	7.7748e+01(2.3260e+01)	5.7167e+01(2.5854e+01)
f_9	1.6138e−02(1.8200e−02)	3.0277e+00(1.6496e+01)	6.0322e+00(2.2881e+01)	3.0270e+00(1.6498e+01)	9.0786e+00(2.7591e+01)
f_{10}	2.5317e+02(4.8562e+02)	5.8473e+02(6.2392e+02)	7.5600e+02(1.0069e+03)	2.0692e+02(4.1388e+02)	4.1767e+02(7.4568e+02)

- For function f_2 and f_9, CBPPSO outperformed the other algorithms. The convergence rate of CBPPSO is more in comparison to other algorithms. The horizontal line shows the inability of the algorithm to further converge towards the optimal solution.
- For function f_3 and f_5, w-PSO and CBPPSO both provided the best solutions. The CBPPSO converges slightly faster than w-PSO towards the optimal solution. Both the algorithm followed by AIWPSO converge must faster towards the optimal solution than the other algorithms.
- For function f_4 and f_{10}, best solution is provided by AIWPSO followed by CBPPSO algorithm. The AIWPSO and CBPPSO converge faster in direction of the best solution than the other algorithms.
- For function f_7, GPSO, Sugeno and w-PSO yield the best solution in comparison to the CBPPSO and AIWPSO. However, CBPPSO converge faster towards the best solution followed by AIWPSO.
- For function f_8, w-PSO outperformed the other algorithms in terms of best solution. The AIWPSO and CBPSO converge in vicinity of the best solution at faster rate than other algorithms.

The outcome shows that CBPPSO yields the best accuracy for 6 out of 10 benchmark problems i.e., for functions f_1,f_2,f_3,f_5,f_6 and f_9. The convergence curve in the logarithmic scale for all the algorithms is shown in Fig. 2 which illustrates the above results. The performance of CBPPSO is comparable in case of functions f_4,f_8 and f_{10}. The algorithms are scored according to number of times the algorithm provides the comparable and best results. The scores of the w-PSO, AIWPSO, Sugeno, GPSO, CBPPSO are 7, 7, 5, 4, and 9 respectively. Most of the time, the best position are discovered by the algorithms with adaptive inertia weight approach. Also these algorithms converges faster towards the optimal solution in comparison to the other time-varying inertia weight based algorithms. In particular, it can be observed that the CBPPSO converges at faster rate towards the best solution in comparison to the other algorithms most of the time.

The proposed approach has also been tested over a real world engineering benchmark problem and the performance is evaluated and compared with other existing algorithms on standard parameters reported in literature. This is discussed in the next section.

6 Real world engineering optimization problem

Apart from the well known benchmark functions, the potential of CBPPSO is also investigated on three real world engineering problems. The swarm size is 20, and the minimum result of CBPPSO is recorded after 10 independent runs, continued for 2500 iterations [15, 31]. For comparison, we have taken results from [12, 31, 32] for problem 1, 2 and 3 respectively as reported in the literature. Best minimum optimization result for the respective problems are indicated in bold in Tables 3, 4 and 5.

6.1 Problem 1: Design of a pressure vessel [33]

The pressure vessel design is a constrained engineering design problem in which the aim is to minimize the cost of manufacturing pressure vessel. The problem is given as follows [12]:

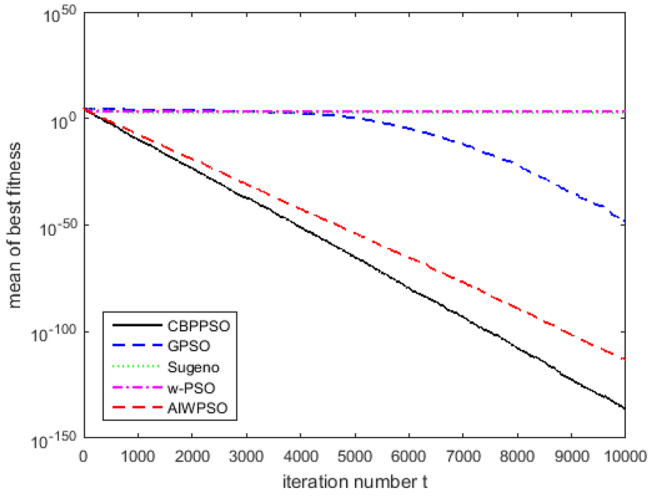
Minimize:

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1611x_1^2x_4 + 19.84x_1^2x_4 + 19.84x_1^2x_3$$

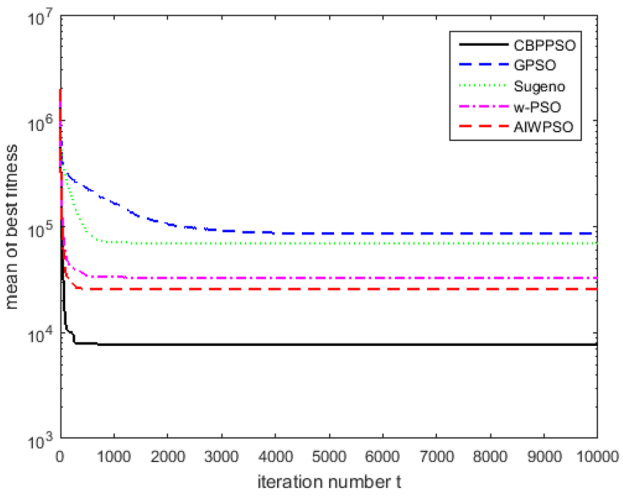
Subject to:

$$g_1(x) = 0.0163x_3 - x_1 \leq 0,$$

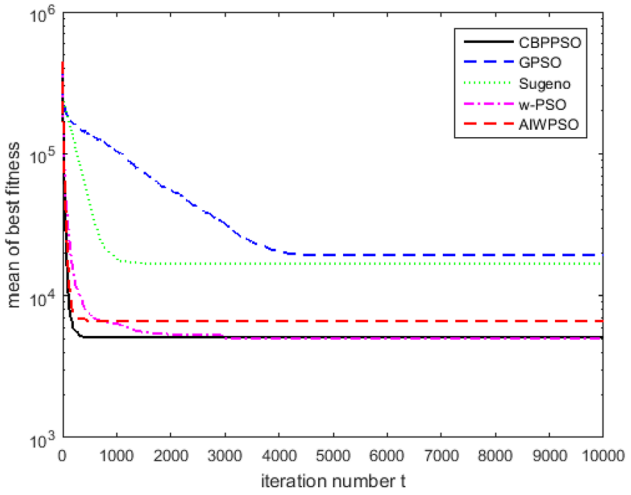
$$g_2(x) = 0.00954x_3 - x_2 \leq 0,$$



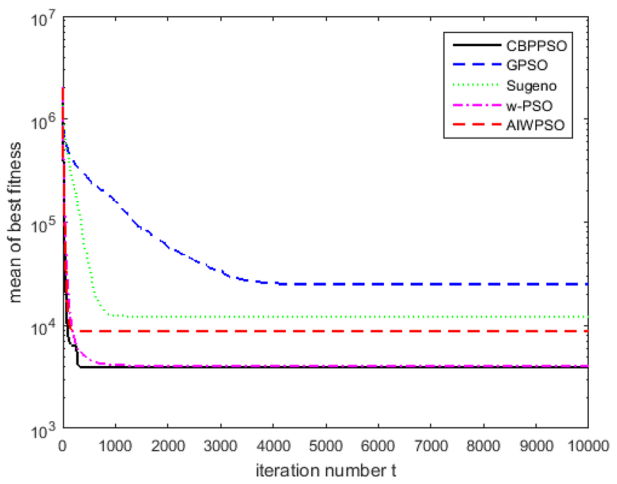
(a) Function f_1



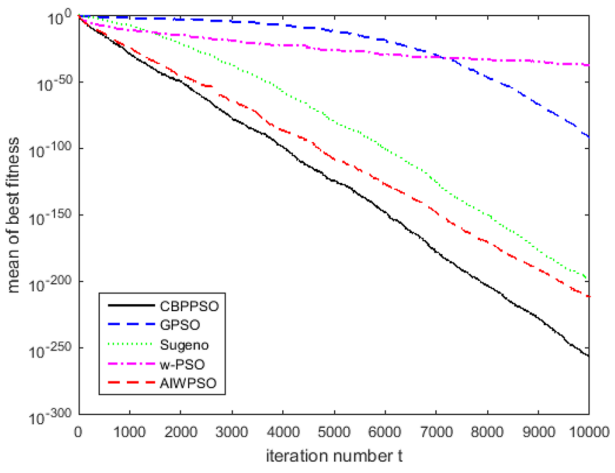
(b) Function f_2



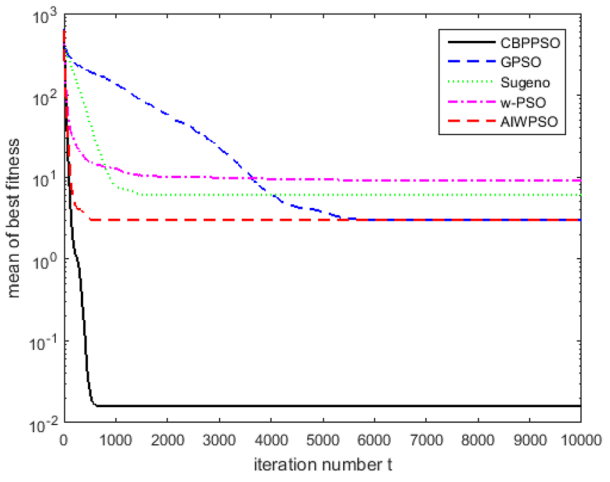
(c) Function f_3



(d) Function f_5



(e) Function f_6



(f) Function f_9

Fig. 2 The best fitness mean for 30 independent realizations as a function of iteration number for functions f_1,f_2,f_3,f_5,f_6 and f_9

$$g_3(x) = 1296000 - \pi x_3^2 x_4 - \frac{4}{3}\pi x_3^2 \leq 0,$$

$$g_4(x) = x_4 - 240 \leq 0,$$

$$g_5(x) = 1.1 - x_1 \leq 0,$$

$$g_6(x) = 0.6 - x_4 \leq 0.$$

The shell thickness $T_s(x_1)$, spherical head thickness $T_h(x_2)$, radius of cylindrical shell $R(x_3)$ and shell length $L(x_4)$ are four design parameters of pressure vessel design problem as shown in Fig. 3. Here x_1 and x_2 are integer multipliers of 0.0625. $x_3 \in [40, 80]$ and $x_4 \in [20, 60]$.

Table 3 summate the optimization results on pressure vessel design problem by CBPPSO, AIWPSO [12], Sandgren [33], CODEQ [35], Harmony search [36], and genetic algorithm [37]. Among the six experimented algorithms, CBPPSO gives the best results.

6.2 Problem 2: Design of a tension/compression spring [39]

The goal in this problem is to minimize the weight of the tension/compression spring, subject to constraint over surge frequency, minimum deflection, shear stress, diameter and design variables. The problem is presented as:

Minimize:

$$f(x) = (x_3 + 2)x_2 x_1^2$$

Subject to:

$$g_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0,$$

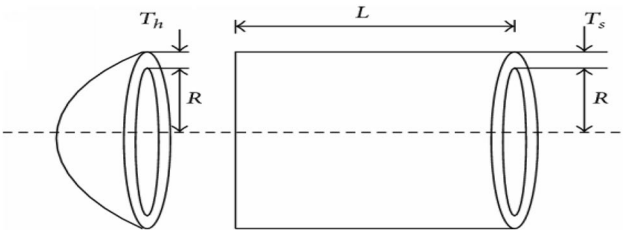


Fig. 3 The pressure vessel problem [34]

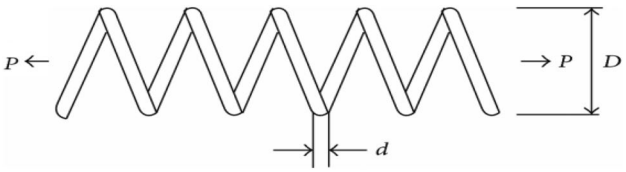


Fig. 4 The tension/compression spring problem [34]

$$g_2(x) = \frac{4x_2^3 - x_1 x_2}{12566\,(x_2 x_1^3 - x_1^4)} - \frac{1}{5108 x_1^2} - 1 \leq 0,$$

$$g_3(x) = 1 - \frac{140.45 x_1}{x_3 x_2^2} \leq 0,$$

$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0.$$

The design variables are the mean coil diameter, $D(x_2)$, the number of active coils, P (x_3), and the wire diameter, d (x_1) as shown in Fig. 4. The design variables have following desired ranges: $0.05 \leq x_1 \leq 2$,$0.25 \leq x_2 \leq 1.3$,and $2 \leq x_3 \leq 15$. Table 4 summates the best results of Belegundu [38], Arora [39], He and Wang [32], Lobato et al. [31], and CBPSO. The CBPPSO provide the better result in comparison to the other approaches.

6.3 Problem 3: Design of a welded beam [39]

The aim in this problem is to minimize the cost of welded beam subject to the constraint over the end deflection of the beam (δ), bending stress in the beam (σ), shear stress (τ), buckling load in the bar (P_C), and side constraints. The problem is presented as:

Minimize:

$$f(x) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14 + x_2)$$

Subject to:

$$g_1(x) = \tau(x) - \tau_{\max} \leq 0,$$

$$g_2(x) = \sigma(x) - \sigma_{\max} \leq 0,$$

Table 3 Pressure design problem optimization results

	Sandgren [33]	Wu and Chow [37]	Harmony search [36]	CODEQ [35]	AIWPSO [12]	CBPPSO
x_1	1.125	1.125	1.125	1.125	1.125	1.125
x_2	0.625	0.625	0.625	0.625	0.625	0.625
x_3	48.97	58.1978	58.2789	58.2901554401	58.2901554404	62.9866
x_4	106.72	44.293	43.7549	43.6926562409	43.6926562409	20
$f(x)$	7980.894	7207.494	7198.433	7197.728	7197.728	**6952.72**

Table 4 Optimization results for tension/compression spring design problem

	Belegundu [38]	Arora [39]	He and Wang [32]	Lobato et al. [31]	CBPPSO
x_1	0.050000	0.053396	0.051728	0.051744	0.0512637
x_2	0.315900	0.399180	0.357644	0.357754	0.3465700
x_3	14.25000	9.185400	11.244543	11.56132	11.909700
$f(x)$	0.012674	0.012730	0.012674	0.012789	**0.0126686**

$$g_5(x) = 0.125 - x_1 \leq 0,$$

$$g_6(x) = \delta(x) - \delta_{\max} \leq 0,$$

$$g_7(x) = P - P_C(x) \leq 0$$

where,

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right),$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},$$

$$\sigma(x) = \frac{6PL}{x_4x_3^2}, \quad \delta(x) = \frac{4PL^3}{Ex_3^3x_4},$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},$$

$$P_C = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),$$

$$P = 6000\,lb, L = 14\,in, E = 3 \times 10^6\,psi, G = 12 \times 10^6\,psi,$$
$$\tau_{\max} = 13{,}600\,psi, \sigma_{\max} = 30{,}000\,psi, \delta_{\max} = 0.25\,in.$$

The problem has four design variables as shown in Fig. 5, i.e., $t(x_3)$, $l(x_2)$, $h(x_1)$, and $b(x_4)$. The variables have following desired ranges : $x_1, x_4 \in [0.1, 2]$, and $x_2, x_3 \in [0.1, 10]$. Table 5 summarizes the best results of Deb [40], Coello [41], He and Wang [32], Coello and Montes [42], and CBPSO. Here also, CBPPSO provide the better result in than the other algorithms.

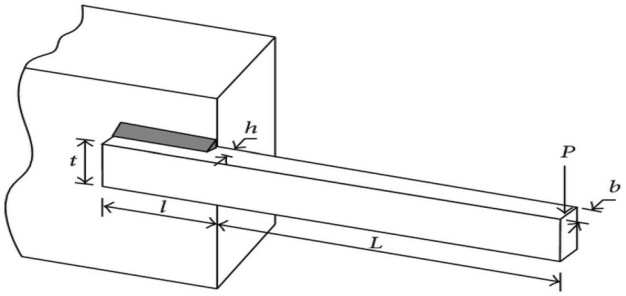


Fig. 5 The welded beam problem [43]

Table 5 Welded beam design problem optimization results

	Deb [40]	Coello [41]	Coello and Montes [42]	He and Wang [32]	CBPPSO
x_1	0.248900	0.208800	0.205986	0.202369	0.20573
x_2	6.173000	3.420500	3.471328	3.544214	3.47049
x_3	8.178900	8.997500	9.020224	9.048210	9.03662
x_4	0.253300	0.210000	0.206480	0.205723	0.20573
$f(x)$	2.433116	1.748309	1.728226	1.728024	**1.72485**

7 Conclusion

The paper introduces a new adaptive inertia weight approach (CBPPSO) which is based on the cumulative binomial probability of the total particles that have improved their position. In this approach, to realize the particles' current state in the search region, the feedback is provided by this cumulative binomial probability and in turn adjusts the inertia weight value accordingly. Experimental results clearly show that the search behavior of the particles is improved due to introduced approach and is more effective in comparison to the other successful variants of PSO. The comparisons and analysis are done in terms of accuracy of the solution and the convergence speed.

Also, we applied CBPPSO to solve three real world engineering problems and compared result with the other approaches. The result provided by CBPPSO is better than outcome of other strategies. Since CBPPSO is simple and efficient, it can be applied to solve the other problems effectively.

References

1. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, Perth, vol 4, pp 1942–1948
2. Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: Evolutionary computation proceedings, IEEE world congress on computational intelligence, The IEEE international conference, Anchorage, pp 69–73
3. Eberhart RC, Shi Y (2001) Tracking and optimizing dynamic systems with particle swarms. In: Evolutionary computation. Proceedings of the IEEE congress, Seoul, vol 1, pp 94–100
4. Shi Y, Eberhart RC (1999) Empirical study of particle swarm optimization. In: Proceedings of the IEEE congress, CEC 99, Washington, DC, vol 3, pp 1945–1950
5. Al-Hassan W, Fayek MB, Shaheen SI (2006) Psosa: an optimized particle swarm technique for solving the urban planning problem. In: International conference on computer engineering and systems, pp 401–405
6. Malik RF, Rahman TA, Hashim SZM, Ngah R (2007) New particle swarm optimizer with sigmoid increasing inertia weight. Int J Comput Sci Secur 1(2):35–44
7. Chen G, Huang X, Jia J, Min Z (2006) Natural exponential inertia weight strategy in particle swarm optimization. In: 6th world congress on intelligent control and automation, vol 1, pp 3672–3675
8. Gao YL, An XH, Liu JM (2008) A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation. In: Computational intelligence and security, IEEE international conference, vol 1, pp 61–65
9. Nikabadi A, Ebadzadeh M (2008) Particle swarm optimization algorithms with adaptive inertia weight: a survey of the state of the art and a novel method. IEEE J Evol Comput
10. Lei K, Qiu Y, He Y (2006) A new adaptive well-chosen inertia weight strategy to automatically harmonize global and local search ability in particle swarm optimization. In: 1st IEEE international symposium on systems and control in aerospace and astronautics, Harbin, pp 977–980
11. Zhan ZH, Zhang J, Li Y, Chung HSH (2009) Adaptive particle swarm optimization. IEEE Trans Syst Man Cybern Part B Cybern 39(6):1362–1381
12. Nickabadi A, Ebadzadeh MM, Safabakhsh R (2011) A novel particle swarm optimization algorithm with adaptive inertia weight. Appl Soft Comput 11(4):3658–3670
13. Kessentini S, Barchiesi D (2015) Particle swarm optimization with adaptive inertia weight. Int J Mach Learn Comput 5(5):368–373
14. Bansal JC, Singh PK, Saraswat M, Verma A, Jadon SS, Abraham A (2011) Inertia weight strategies in particle swarm optimization. In: Nature and biologically inspired computing (NaBIC), third world congress, pp 633–640
15. Jordehi AR, Jasni J (2015) Particle swarm optimization for discrete optimization problems: a review. Artif Intell Rev 43(2):243–258
16. Wang D, Tan D, Liu L (2018) Particle swarm optimization algorithm: an overview. Soft Comput 22(2):387–408
17. Ghamisi P, Benediktsson JA (2015) Feature selection based on hybridization of genetic algorithm and particle swarm optimization. IEEE Geosci Remote Sens Lett 12(2):309–313
18. Momeni E, Armaghani DJ, Hajihassani M, Amin MFM (2015) Prediction of uniaxial compressive strength of rock samples using hybrid particle swarm optimization-based artificial neural networks. Measurement 60:50–63
19. Gao Y, Du W, Yan G (2015) Selectively-informed particle swarm optimization. Sci Rep 5:9295
20. Satapathy SC, Chittineni S, Krishna SM, Murthy JVR, Reddy PP (2012) Kalman particle swarm optimized polynomials for data classification. Appl Math Model 36(1):115–126
21. Tian D, Shi Z (2018) MPSO: modified particle swarm optimization and its applications. Swarm Evol Comput 41:49–68
22. Agrawal A et al (2018) Particle swarm optimization with probabilistic inertia weight. In: Harmony search and nature inspired optimization algorithms, ICHSA 2018, pp 239–248
23. Surjanovic S, Bingham D (2013) Virtual library of simulation experiments: test functions and datasets [online]. http://www.sfu.ca/~ssurjano/. Accessed Dec 2017
24. Molga M, Smutnicki C (2005) Test functions for optimization needs. http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf. Accessed Dec 2017
25. Global optimization test problems. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO.htm. Accessed Dec 2017
26. Dixon LCW, Szego GP (1978) The global optimization problem: an introduction. Towards Glob Optim 2:1–15
27. Back T (1996) Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press, New York
28. Laguna M, Marti R (2002) Experimental testing of advanced scatter search designs for global optimization of multimodal functions. http://www.uv.es/rmarti/paper/docs/global1.pdf. Accessed Dec 2017
29. Global optimization test functions index. http://infinity77.net/global_optimization/test_functions.html#test-functions-index. Accessed Dec 2017
30. Martínez JF, Gonzalo EG (2009) The PSO family: deduction, stochastic analysis and comparison. Swarm Intell 3(4):245–273
31. Lobato FS, Steffen Jr V (2014) Fish swarm optimization algorithm applied to engineering system design. Latin Am J Solids Struct 11(1):143–156
32. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20(1):89–99
33. Sandgren E (1990) Nonlinear integer and discrete programming in mechanical design optimization. J Mech Des 112(2):223–229
34. Dong M, Wang N, Cheng X, Jiang C (2014) Composite differential evolution with modified oracle penalty method for constrained optimization problems. Math Probl Eng 2014:617905
35. Omran MG (2010) CODEQ: an effective metaheuristic for continuous global optimization. Int J Metaheuristics 1(2):108–131
36. Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng 194(36):3902–3933
37. Wu SJ, Chow PT (1995) Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization. Eng Optim 24(2):137–159
38. Belegundu AD (1982) A study of mathematical programming methods for structural optimization. Dept. of Civil Environ. Eng., Iowa Univ
39. Arora JS (1989) Introduction to optimum design. McGraw-Hill, New York
40. Deb K (1991) Optimal design of a welded beam via genetic algorithms. AIAA J 29(11):2013–2015
41. Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41(2):113–127
42. Coello CAC, Montes EM (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Inform 16(3):193–203
43. Zheng H, Zhou Y (2013) A cooperative coevolutionary cuckoo search algorithm for optimization problem. J Appl Math 2013:912056