Particle Swarm Optimization with Dynamic Inertia Weights

Yogendra Singh Kushwah and R K Shrivastava

Department of Mathematics, Government Science College, Gwalior, MP, India.

***Abstract*—Particle swarm optimization (PSO) is a popular swarm intelligence based optimization algorithm. Inertia weight is an important parameter of PSO. Many strategies have been proposed for setting of inertia weight during last** 20 **years. Dynamic Inertia Weight approach is proposed in this paper. In the proposed approach for inertia weights are applied to PSO based on probability. Numerical experiments on** 25 **test problems indicate that the proposed approach of Dynamic Inertia weights has the potential to improve the performance of PSO.**

***Keywords*—Particle swarm optimization, Swarm Intelligence, Inertia weight strategies, Convergence.**

I. INTRODUCTION

Inertia weight is an important parameter of PSO. It has the capacity of maintain to balance between exploration and exploitation. Inertia weight is used to provide the optimum velocity to the particle at the present time stamp. When first version of PSO presented by Eberhart and Kennedy in 1995 [1], then there was no concept of Inertia Weight given by him. First time Inertia Weight introduced by Shi and Eberhart [2] in 1998, with constant inertia weight. They proposed to higher and small Inertia Weight which helps in global and local search. Further there are many Inertia Weight strategies proposed by researchers that give batter performance of PSO for different type of Optimization problems. A brief review of Inertia Weight strategies that is used in PSO are explained in subsequent Sections. Bansal et al. [3] did work with different strategies of Inertia Weight. The main purpose of this study is that selecting the best inertia weight strategies. Table I shows different inertia weight strategies which is required in DIWPSO. In this paper rate of change of particle position can be updated through control parameter D which choose inertia weight from four inertia weight strategies on basis of equal probability. DIWPSO Is tested over 25 linear, non linear, single variable, multivariate, constrained optimization test problems taken from literature. The results analysis shows that the modified PSO is better than basic PSO with respect to accuracy, rate of convergence and robustness.

Organization of the paper given as: A little description about Particle Swarm Optimization given in section II. Section III provides various Inertia Weight strategies which used in PSO. Section IV Focuses on basic PSO with dynamic inertia weight strategies. Section V highlights on results. Section VI covered conclusion.

II. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization is a swarm intelligence-based optimization technique. Kennedy and Eberhart [1] was given this technique in 1995.This technique based on collective social response in a group which shown in some Insects or Animals. Such type of behavior can be observe in fish schooling and bird flocking. In PSO, every individual is known as particle that is involved in the searching of food source (optimal solution). Every particle has the capability to remember own *pbest* solution which gained during searching the *gbest* solution. Every, i^{th} particle is described as $y_i = (y_{i1}, y_{i2}, \ldots, y_{in})$. Randomly generate the population is an initial solutions which search optimal solution. The *pbest* is the personal best position of an individual in the swarm. The *gbest* is the global best position among all the individual in the swarm. The i^{th} individual best position is represented as $p_i = (p_1, p_2, \ldots, p_n)$. The velocity of a particle is the rate of change in the position and represented as $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$. The particles are update the velocity and position through following equations:

$$v_{id} = w^* v_{id} + c_1 * r_1 (p_{id} - y_{id}) + c_2 * r_2 (p_{gd} - y_{id}) \quad (1)$$

$$y_{id} = y_{id} + v_{id} \quad (2)$$

where w, c_1, c_2, r_1 and r_2 are parameters. $d = \{1, 2, 3, \ldots, N\}$, w represents the inertia weight which provides equivalence in the activity of exploration and exploitation in the search space. c_1, c_2, are acceleration constants which enforce to each individual in the swarm to attain personal best and global best position. r_1 and r_2 represent random number which get the value in the interval $[0, 1]$. If put the large value of r_1 and r_2 then particle search speed is fast, in this situation particle can miss the *gbest* position. Small value of r_1 and r_2 breakdown the speed of the particle and may be he gets pre mature solution. v is the velocity of the individual which lies between v_{min} and v_{max}. The range of the velocity is constrained of the individual in the solution space. Very small velocity might miss the opportunity of the particle to search global best solution and very high velocity might be unable to reach Optimal solution. The pseudo code of PSO algorithm is given in Algorithm (1).

Algorithm 1: Basic Particle Swarm Optimization

Input:

Initialize the parameters

Begin:

for individual **do**

initialize the swarm.

end for

iter=0

while termination criteria is not met **do**

for individual **do**

Evaluate fitness value.

if new fitness value is better than *pbest* **then**

pbest = new calculated fitness value

end if

end for

Find *gbest* from the *pbest* particles.

for each particle **do**

Evaluate velocity of particle from (1).

Evaluate position of particle from (2).

end for

iter = iter + 1

end while

III. STRATEGIES OF INERTIA WEIGHT FOR PARTICLE SWARM OPTIMIZATION

Inertia weight is an important parameter of PSO. It has the capacity of maintain to balance between exploration and exploitation. Inertia weight is used to provide the optimum velocity to the particle at the present time stamp. When first version of PSO presented by Eberhart and Kennedy in 1995 [1], then there was no concept of Inertia Weight given by him. First time of Inertia Weight introduced by Shi and Eberhart [2] in 1998, with constant inertia weight. They proposed to higher and small Inertia Weight which helps in global and local search. Further there are many Inertia Weight strategies proposed by researchers that give batter performance of PSO for different type of Optimization problems. A brief review of Inertia Weight strategies that is used in PSO are explained in subsequent Sections.

Bansal et al. [3] did work with different strategies of Inertia Weight. The main purpose of this study is that selecting the best inertia weight strategies. Table I shows different inertia weight strategies which are required in DIWPSO. In this paper rate of change of particle position can be updated through control parameter D which choose inertia weight from four inertia weight strategies on basis of equal probability. DIWPSO is tested over 25 linear, nonlinear, single variable, multivariable, constrained and non constrained optimization test problems taken from literature. The results analysis shows that the modified PSO is better than basic PSO with respect to accuracy, rate of convergence and robustness. Chen et al. [4] proposed two different Inertia Weight strategies which based on Decreasing Inertia Weight. It is also analysed that proposed strategies is better than with respect to convergence of solution in early stage of approximation. It is also found that proposed strategies improve the result for continuous optimization problem. Fayek et al. [5] presented (PSOSA) which based on hybridize the Particle Swarm with Simulated Annealing, it was performed on urban planning problem and result shows that rate of convergence is higher the previous one. Increased load of growing number of blocks in this problem are also adaptive.

TABLE I: Selected Inertia Weight [3]

SN	Inertia Weight strategies	Formula of Inertia Weight
1.	Constant Inertia Weight [2]	$w = c$ $c = 0.7$ (considered for experiments)
2.	Random Inertia Weight [10]	$w = 0.5 + \text{rand}()/2$
3.	Chaotic Inertia Weight [6]	$z = 4 * z * (1 - z)$ $w = (w_1 - w_2) * \frac{MAX_{iter} - iter}{MAX_{iter}} + w_2 * z$
4.	Linear decreasing Inertia Weight [11]	$w_k = w_{max} - \frac{w_{max} - w_{min}}{iter_{max}} * k$

Fayek et al. [5] hybridize the Particle Swarm with Simulated Annealing (PSOSA) in which Inertial Weight was optimized and it was tested on urban planning problem and it found that the hybrid approach is better in respect to fast rate of convergence. It is also adaptive in urban planning problem to increased load of growing number of blocks. Two Natural Exponent Inertia Weight strategies were proposed Chen et al. [4], which are based on the Decreasing Inertia Weight. It is found that the proposed strategies converge faster than linear one during the early stage of iteration. Also it provides better results for most continuous optimization problems. Chaotic Inertia Weight was proposed by Feng et al. [6]. It uses the merits of chaotic optimization. It is found that CRIW PSO performs better that RIWPSO. Malik et al. [7] proposed Sigmoid Increasing Inertia Weight strategy (SIIW) which has capability to fast convergence when fitness value of the function is minimum. In this approach Linear Increasing Inertia Weight combine with sigmoid function. To increase the overall convergence speed Gao et al. [8] proposed (LDIW), this is the Chaos mutation operator which was combined with Logarithm Decreasing Inertia Weight. The Chaos mutation increase the ability to come out of the local minima a while LDW improves its convergence speed. Exponent Decreasing Inertia Weight (EDIW) was also proposed by Gao et al. [8] that overcome premature convergence. Also stochastic mutation [9] with Exponent Decreasing Inertia Weight improves the performance of PSO and getting out from local minima.

IV. DIWPSO: DYNAMIC INERTIA WEIGHTS BASED PARTICLE SWARM OPTIMIZATION

Bansal et al. [3] did work with different strategies of Inertia Weight. The main purpose of this study is that selecting the best inertia weight strategies. Table I shows different inertia weight strategies which are required in DIWPSO. In this paper rate of change of particle position can be updated through control parameter D which choose inertia weight from four inertia weight strategies on the basis of equal probability. DIWPSO is tested over 25 linear, nonlinear, single variable, multivariate, separable and non-separable optimization test problems taken from literature. The results analysis shows that the modified

PSO is better than basic PSO with respect to accuracy, rate of convergence and robustness.

The algorithm (2) is the pseudo of DIWPSO.

Algorithm 2: Dynamic Inertia Weight Particle Swarm Optimization (DIWPSO)

Input:
Initialize the parameters
DIW* = [CoIW, RIW, ChIW, LDIW]
*ChIW = Chaotic Inertia Weight; RIW = Random Inertia Weight; CoIW = Constant Inertia Weight;
LDIW = Linear decreasing Inertia Weight
w = Linear decreasing Inertia Weight
Output:
(gbest) particle
Begin:
for individual **do**
initialize the swarm.
end for
iter=0
while termination criteria is not satisfied **do**
for individual **do**
Evaluate fitness value for each particle.
if Evaluated value is better than *pbest* **then**
pbest = Evaluated fitness value
end if
end for
find *gbest* form the *pbest* particles.
for each particle **do**
Choose inertia weight (w) using following strategy.
If rand() < D **then**
i = **uniformly distributed random Integer in (1,4)** \\
w = **DIW [i]**
end if
Evaluate velocity of particle from (1).
Evaluate position of particle from (2).
end for
iter = iter + 1
end while

V. RESULTS ANALYSIS AND DISCUSSION

To proposed the Dynamic Inertia Weights strategy in PSO, consider 25 optimization test problems for experiments.

A. Experimental Settings

The Population size is 50. The number of decision variables is assumed to be 10. The function evaluations, for termination, is set to be 200, 000. Initially 100 simulations are taken to avoid the effect of choice of initial population. The acceleration parameters, c_1 and c_2 are set to 2. To test the performance of the Proposed algorithm, 25 different test optimization problems are used for experiments. All the used test problems are shown in Table III with their search space range. The proposed algorithm was implemented in C++ and compiled in GCC compiler.

B. Experimental Results and Analysis

The results of PSO and DIWPSO are analyzed on the basis of three criteria average mean error, average mean function evaluation and success rate. In Figure 1, the proposed algorithm DIWPSO is tested with

Different settings of D, ranging term 0.1 to 1. It is observed that the most optimum choice of parameter D is 0.7. The modified technique is tested over 25 optimization problems. The test problems are given in Table III. The numerical results are shown in Table IV. In Table III, standard deviation (SD), mean error (ME), average function evaluations (AFE) and success rate (SR) are reported. It can be observed from Table IV, that DIWPSO is better in terms of efficiency, accuracy and reliability.

To verify the significance difference between data, Mann-Whitney U rank sum test is carried out. In this study, this test is performed at 5 percentage level of significance between the data sets of DIWPSO and PSO. The data sets are the average function evaluations (AFE) and the null hypothesis is defined as: 'There is no significance difference between data'. In Table II, '+' shows improvement by DIWPSO while '-' shows no changes in DIWPSO with respect to basic PSO. Table II shows that the DIWPSO is better than PSO while it is checked on 25 problems. In Figure 2, analyse the performance of DIWPSO with respect to PSO. In Bar-Diagram clear shows that DIWPSO outperformed in three different criteria that is average mean error, average mean function evaluation and success rate.

TABLE II: The comparison of DIWPSO and PSO.

Test Problem	DIWPSO	Test Problem	DIWPSO
g_1	+	g_{14}	+
g_2	-	g_{15}	+
g_3	+	g_{16}	-
g_4	+	g_{17}	-
g_5	-	g_{18}	+
g_6	+	g_{19}	+
g_7	-	g_{20}	+

g_8	+	g_{21}	-
g_9	+	g_{22}	+
g_{10}	+	g_{23}	+
g_{11}	-	g_{24}	+

g_{12}	+	g_{25}	+
g_{13}	-		

TABLE III: Test problems, AE: Acceptable Error, D: Dimension of the Problem, C': Characteristics, U: Unimodal, M: Multimodal, S: Separable, NS: Non-separable

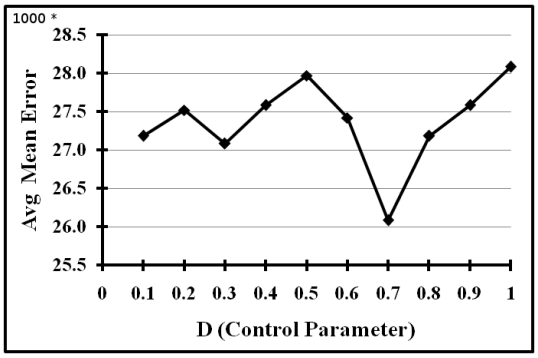
SN	Test Problem	Objective function	Search Range	Optimum Value	D	C'	AE
1	Exponential	$g_1(y) = -(\exp(-0.5\sum_{i=1}^{D} y_i^2)) + 1$	[-1 1]	$g(\vec{0}) = -1$	30	M, S	1.0E-05
2	Cigar	$g_2(y) = y_0^2 + 100000\sum_{i=1}^{D} y_i^2$	[-10 10]	$g(\vec{0}) = 4$	30	U, S	1.0E-05
3	Sphere	$g_3(y) = \sum_{i=1}^{D} y_i^2$	[-5.12 5.12]	$g(\vec{0}) = 0$	30	U, S	1.0E-05
4	Rastrigin	$g_4(y) = 10D + \sum_{i=1}^{D}[y_i^2 - 10\cos(2\pi y_i)]$	[-5.12 5.12]	$g(\vec{0}) = 0$	30	M, S	1.0E-05
5	Salomon Problem	$g_5(y) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{D} y_i^2}\right) + 0.1(\sqrt{\sum_{i=1}^{D} y_i^2})$	[-100 100]	$g(\vec{0}) = 0$	30	M, S	1.0E-01
6	Schewel	$g_6(y) = \sum_{i=1}^{D} \lvert y_i\rvert + \prod_{i=1}^{D} \lvert y_i\rvert$	[-10 10]	$g(\vec{0}) = 0$	30	M, NS	1.0E-05
7	brown3	$g_7(y) = \sum_{i=1}^{D-1}(y_i^{2(y_{i+1}^2)+1} + y_{i+1}^{2(y_i^2)+1})$	[-1 4]	$g(\vec{0}) = 0$	30	U, NS	1.0E-05
8	Axis parallel hyper-ellipsoid	$g_8(y) = \sum_{i=1}^{D} i * y_i^2$	[-5.12 5.12]	$g(\vec{0}) = 0$	30	U, S	1.0E-05
9	Neumaier 3 Problem (NF3)	$g_9(y) = \sum_{i=1}^{D} y_{i-1}^2 - \sum_{i=2}^{D} y_i y_{i-1}$	[$-D^2$ D^2]	$g(\vec{0}) = -(D(D+4)(D-1))/6.0$	10	M, NS	1.0E-01
10	Beale function	$g_{10}(y) = [1.5 - y_1(1-y_2)]^2 + [2.25 - y_1(1-y_2^2)]^2 + [2.625 - y_1(1-y_2^3)]^2$	[-4.5 4.5]	$g(3, 0.5) = 0$	2	U, NS	1.0E-05
11	Ellipsoidal	$g_{11}(y) = \sum_{i=1}^{D}(y_i - i)^2$	[-D D]	$g(1,2,..D) = 0$	30	U, S	1.0E-05
12	Rotated hyper-ellipsoid	$g_{12}(y) = \sum_{i=1}^{D}\sum_{j=1}^{i} y_j^2$	[-65.536, 65.536]	$g(\vec{0}) = 0$	30	U, S	1.0E-05
13	Colville function	$g_{13}(y) = 100\,(y_2 - y_1^2)^2 + (1-y_1)^2 + 90(y_4 - y_3^2)^2 + (1-y_3)^2 + 10.1[(y_2-1)^2 + (y_4-1)^2] + 19.8(y_2-1)(y_4-1)$	[-10 10]	$g(\vec{1}) = 0$	4	M, NS	1.0E-05
14	Braninss function	$g_{14}(y) = a\,(y_2 - by_1^2 + cy_1 - d)^2 + e(1-f)\cos y_1 + e$	$y_1 \in [-5\ 10]$, $y_2 \in [0\ 15]$,	$g(-\pi, 12.275) = 0.3979$	2	U, NS	1.0E-05
15	Kowalik	$g_{15}(y) = \sum_{i=1}^{11}[a_i - \frac{y_1(b_i^2 + b_i y_2)}{b_i^2 + b_i y_3 + y_4}]^2$	[-5 5]	3.07486E-04	4	M, NS	1.0E-05
16	Shifted Rosenbrock	$g_{16}(y) = \sum_{i=1}^{D-1}(100\,(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + g_{bias}$ $z = y - 0 + 1,\quad y = [y_1, y_2, \dots, y_D],\quad o = [o_1, o_2, \dots, o_D]$	[-100 100]	$g(0) = g_{bias} = 390$	10	U, NS	1.0E-01
17	Shifted Sphere	$g_{17}(y) = \sum_{i=1}^{D}(z_i^2) + g_{bias},\quad z = y - 0,$ $y = [y_1, y_2, \dots, y_D],$ $o = [o_1, o_2, \dots, o_D]$	[-100 100]	$g(0) = g_{bias} = -450$	10	M, S	1.0E-05

18	Shifted Rastrigin	$g_{18}(y) = \sum_{i=1}^{D}[z_i^2 - 10\cos(2\pi z_i) + 10] + g_{bias}$, $z = y - 0$, $y = [y_1, y_2, \dots, y_D]$, $o = [o_1, o_2, \dots, o_D]$	[-5 5]	$g(0) = g_{bias} = -330$	10	U, S	1.0E-02
19	Shifted Schwefel	$g_{19}(y) = \sum_{i=1}^{D}(\sum_{j=1}^{i} z_j)^2 + g_{bias}$, $z = y - 0$, $y = [y_1, y_2, \dots, y_D]$, $o = [o_1, o_2, \dots, o_D]$	[-100 100]	$g(0) = g_{bias} = -450$	10	M, NS	1.0E-05
20	Shifted Ackley	$g_{20}(y) = -20(\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)) + 20 + e + g_{bias}$, $z = y - 0$, $y = [y_1, y_2, \dots, y_D]$, $o = [o_1, o_2, \dots, o_D]$	[-32 32]	$g(0) = g_{bias} = -140$	10	M, S	1.0E-05
21	Goldstein-Price	$g_{21}(y) = (1 + (y_1 + y_2 + 1)^2 + (19 - 14y_1 + 3y_1^2 - 14y_2 + 6y_1y_2 + 3y_2^2)(30 + (2y_1 - 3y_2)^2(18 - 32y_1 + 12y_1^2 + 48y_2 - 36y_1y_2 + 27y_2^2))$	[-2 2]	$g(0, -1) = 3$	2	U, NS	1.0E-14
22	Six-hump camel back	$g_{22}(y) = \left(4 - 2.1y_1^2 + \frac{y_1^4}{3}\right)y_1^2 + y_1y_2 + (-4 + 4y_2^2)y_2^2$	[-5 5]	$g(-0.0898, 0.7126) = -1.0316$	2	M, NS	1.0E-05
23	Easom's function	$g_{23}(y) = -cosy_1cosy_2e^{((-(y_1-\pi)^2-(y_2-\pi)^2))}$	[-10 10]	$g(\pi, \pi) = -1$	2	M, NS	1.0E-13
24	Shubert	$g_{24}(y) = -\sum_{i=1}^{5} i\cos((i+1)y_1 + 1)\sum_{i=1}^{5} i\cos((i+1)y_2 + 1)$	[-10 10]	$g(7.0835, 4.8580) = -186.7309$	2	M, NS	1.0E-05
25	McCormick	$g_{25}(y) = \sin(y_1 + y_2) + (y_1 - y_2)^2 - \left(\frac{3}{2}\right)y_1 + \left(\frac{5}{2}\right)y_2 + 1$	$y_1 \in [-1.5\ 4]$, $y_2 \in [-3\ 3]$,	$g(-0.547, -1.547) = -1.9133$	30	U, NS	1.0E-04

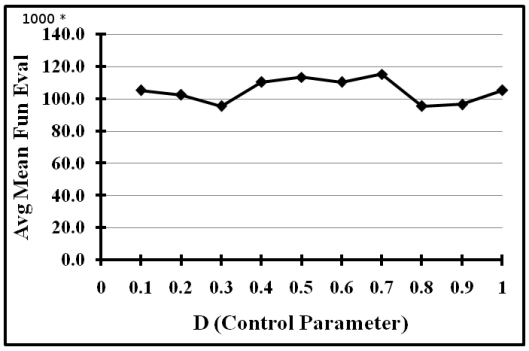
TABLE IV: Compare of DIWPSO with PSO on the basis of benchmark test problems for standard deviation (SD), mean error(ME), average function evaluations (AFE) and success rate (SR).

Test Problems	**PSO**				**DIWPSO**			
	SD	**ME**	**AFE**	**SR**	**SD**	**ME**	**AFE**	**SR**
g_1	0.132630825	6.65E − 03	295063	35	0.63582886	3.14E − 02	294892.8	35
g_2	22.65206365	1.11E + 01	30200	0	5.20607990	2.60E + 01	30200	0
g_3	7.72E − 06	3.63E − 07	53425	100	1.04E − 06	5.13E − 07	53156.9	100
g_4	0.00156257	7.83E − 04	85200	96	0.00609352	4.60E − 04	77825	99
g_5	4.56E − 07	2.20E − 07	102732	100	5.89E − 07	2.98E − 07	101015.2	100
g_6	0.020426675	1.04E − 02	235885	59	0.01853064	9.67E − 03	180162.9	67
g_7	0.034187542	1.73E − 02	298219	5	0.03781919	1.81E − 02	296239.6	7
g_8	0.416263257	2.05E − 01	234685	66	0.32011238	1.60E − 02	220526.4	71
g_9	7.20E − 07	4.02E − 06	62789	81	6.04E − 06	3.42E − 06	36785.3	88
g_{10}	0.000333225	1.65E − 04	55534	82	0.00027267	1.35E − 04	38125	92

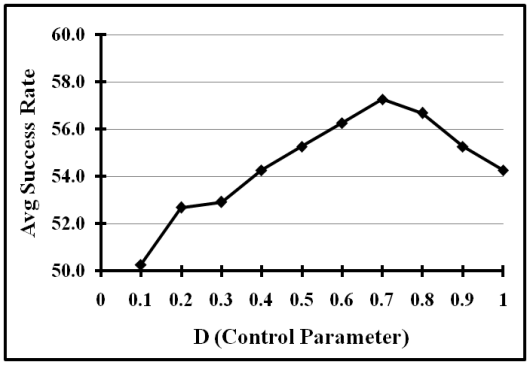
g_{11}	1.32E − 06	6.84E − 07	18826	100	1.65E − 06	7.82E − 07	18643	100
g_{12}	15.35514103	7.64E + 00	300010	0	11.7703564	5.65E + 00	300030	0
g_{13}	5868.778862	2.90E + 03	300000	0	3986.10606	1.98E + 03	300030	0
g_{14}	0.009648218	4.83E − 03	225122	69	0.00309219	1.54E − 03	166692.6	92
g_{15}	9.40E − 06	4.72E − 07	28596	100	7.98E − 07	3.96E − 08	29465	100
g_{16}	0.006709735	3.34E − 03	44458	100	0.00606627	3.02E − 04	44986.8	100
g_{17}	4.76E − 14	2.26E − 14	158868	46	4.81E − 14	2.26E − 15	153214	52
g_{18}	2.52E − 05	1.22E − 06	285082	6	3.82E − 06	1.84E − 07	261213.2	12
g_{19}	0.158572268	7.92E − 02	300000	0	0.21469312	1.06E − 02	292244.8	23
g_{20}	8.24E − 16	4.14E − 17	141010	100	1.03E − 16	5.14E − 18	138052.8	100
g_{21}	3.68E − 04	1.66E − 04	150992	52	3.52E − 04	1.61E − 06	114414.2	65
g_{22}	0.576294692	2.86E − 01	275672	63	0.62532912	3.12E − 02	274586.5	64
g_{23}	4.991971306	2.52E − 00	104838	78	4.64655116	2.30E − 00	98872.7	82
g_{24}	1.345343246	4.51E − 01	44827	86	2.34323344	1.11E − 02	87652.4	88
g_{25}	0.343456678	0.52E − 02	55007	58	2.61123232	0.30E − 04	23452.7	72



(a) Average Mean Error

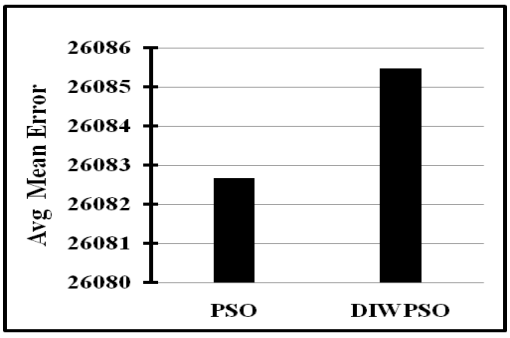


(b) Average Mean Function Evaluation

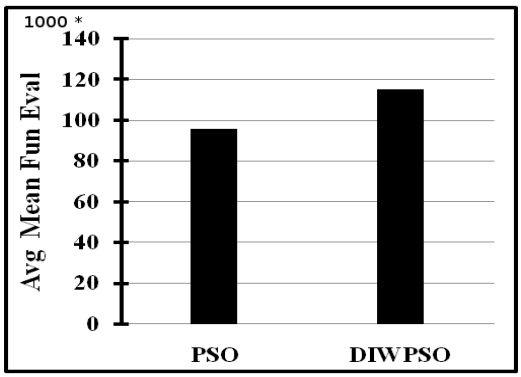


(c) Average Success Rate

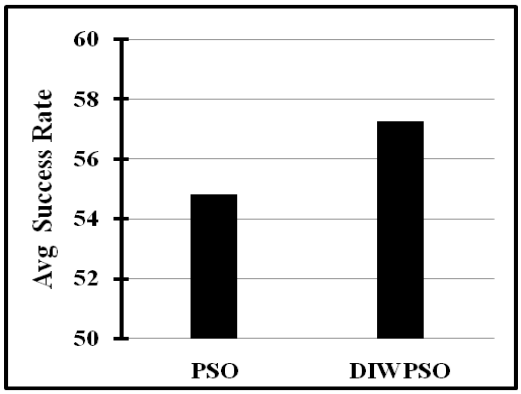
Fig. 1: Evaluate D based on (AE), (AFE) and (SR)



(a) Average Mean Error



(b) Average Mean Function Evaluation



(c) Average Success Rate

Fig. 2: Comparison of PSO and DIWPSO based on (AE), (AFE) and (SR)

VI. CONCLUSION

Modified PSO (DIWPSO) presented in This paper. DIWPSO performed with control parameter D. D selects Inertia Weight with equal probabilities among four Inertia Weight strategies in each approximation. Proposed Algorithm DIWPSO tested over 25 different Optimization Problems. It is found that DIWPSO gives better result with regards to average mean error, average mean function evaluation and success rate. In Future it is possible that more effective Inertia Weight strategies will produce higher accuracy in result.

REFERENCES

[1]. R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pages 39–43. Ieee, 1995.

[2]. Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73. IEEE, 1998.

[3]. JC Bansal, PK Singh, Mukesh Saraswat, Abhishek Verma, Shimpi Singh Jadon, and Ajith Abraham. Inertia weight strategies in particle swarm optimization. In *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on*, pages 633–640. IEEE, 2011.

[4]. G. Chen, X. Huang, J. Jia, and Z. Min. Natural exponential inertia weight strategy in particle swarm optimization. In *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, volume 1, pages 3672–3675. IEEE, 2006.

[5]. W Al-Hassan, MB Fayek, and SI Shaheen. Psosa: An optimized particle swarm technique for solving the urban planning problem. In *Computer Engineering and Systems, The 2006 International Conference on*, pages 401–405. IEEE, 2006.

[6]. Y. Feng, G.F. Teng, A.X. Wang, and Y.M. Yao. Chaotic Inertia Weight in Particle Swarm Optimization. In *Innovative Computing, Information and Control, 2007. ICICIC'07. Second International Conference on*, page 475. IEEE, 2008.

[7]. R.F. Malik, T.A. Rahman, S.Z.M. Hashim, and R. Ngah. New Particle Swarm Optimizer with Sigmoid Increasing Inertia Weight. *International Journal of Computer Science and Security (IJCSS)*, 1(2):35, 2007.

[8]. Y. Gao, X. An, and J. Liu. A Particle Swarm Optimization Algorithm with Logarithm Decreasing Inertia Weight and Chaos Mutation. In *Computational Intelligence and Security, 2008. CIS'08. International Conference on*, volume 1, pages 61–65. IEEE, 2008.

[9]. H.R. Li and Y.L. Gao. Particle Swarm Optimization Algorithm with Exponent Decreasing Inertia Weight and Stochastic Mutation. In *— 2009 Second International Conference on Information and Computing Science*, pages 66–69. IEEE, 2009.

[10]. R.C. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 94–100. IEEE, 2002.

[11]. Jianbin Xin, Guimin Chen, and Yubao Hai. A particle swarm optimizer with multi-stage linearly-decreasing inertia weight. In *Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on*, volume 1, pages 505–508. IEEE, 2009.