Life Science Journal

Websites:
http://www.lifesciencesite.com
http://www.sciencepub.net

Emails:
editor@sciencepub.net
sciencepub@gmail.com

MARSLAND PRESS
Multidisciplinary Academic Journal Publisher

# A New Hybrid Particle Swarm Optimization-Differential Evolution Algorithm

Fariaa Hameed [1,2], Siti Mariyam Hj. Shamsuddin[1]

[1] Soft Computing Research Group, Faculty of Computing, Universiti Teknologi Malaysia, Skudai, 81310 Johor, Malaysia;

[2] Computer Science Institute, Sulaimani Polytechnique University, Sulimanyah, Kurdistan, Iraq

fariaa1980@yahoo.com

**Abstract:** Particle Swarm Optimization (PSO) is one of the population-based heuristic searching algorithms. PSO has good ability of getting global optimization. However, there are some weaknesses such as, low convergence accuracy and speed, similar to other Evolutionary Algorithms (EA). Due to this, a large number of approaches have been developed to improve its performance in recent years, including such changes as hybrid models. In this paper we present a hybrid optimization method based on Particle Swarm Optimization and Differential Evolution. Under this method, we enhance the exploration ability of the particle, by using DE abilities at each iteration step to get best solution. The proposed method is assessed using a set of multimodal functions and real life problems benchmarks.
[Fariaa H., Shamsuddin M.Hj. **A New Hybrid Particle Swarm Optimization-Differential Evolution Algorithm.** . *Life Sci J* 2021;18(6):20-29] (ISSN:1097-8135). http://www.lifesciencesite.com. 3.doi:10.7537/marslsj180621.03.

**Keywords:** Optimization; Particle Swarm Optimization; Differential evolution; Hybrid Particle Swarm Optimization Differential Evolution

## 1. Introduction

One of the most important ethics in our world is the search for a best state. As long as humankind exists, we strive for perfection in many areas. We want to reach a maximum degree of happiness with the least amount of effort. In our economy, profit and sales must be maximized and costs should be as low as possible. Therefore, optimization is one of the oldest of sciences which even extends into daily life. If something is important, general, and abstract enough, there is always a mathematical discipline dealing with it. Global optimization is the branch of applied mathematics and numerical analysis that focuses on, well, optimization. Therefore optimization problems are very important especially in the industrial applications and also in the scientific field. Conventional optimization algorithms involve classical approaches such as dynamic programming, branch-and-bound, and gradient-based methods, while modern optimizations deal with meta-heuristics searching. The examples of meta-heuristics algorithms are simulated annealing, evolutionary computation EC, and ant colony optimization etc.

In recent years it has become clearer that the focusing is on meta-heuristic algorithm is quite limited. The ability of combination of meta-heuristic with other optimization techniques is called hybrid meta-heuristic. These techniques provide more efficient behavior and great flexibility when dealing with real-world and large scale problems. For these reasons, hybrid meta-heuristics currently enjoy an increasing interest in the optimization community. Among the existing meta-heuristic algorithms, a well-known branch is the Particle Swarm Optimization (PSO) which is a stochastic search procedure based on observations of social behaviors of animals, such as bird flocking and fish schooling. Since this type of algorithm simultaneously evaluates many points in the search space and can utilize global information, it is more likely to find the global solution of a given problem. Currently, PSO has been successfully applied to optimizing various continuous nonlinear functions in practice [1]. Hybridization of PSO with local search has been investigated in many studies [2] but only a few algorithms combining it with other global optimization techniques have been proposed and hardly anyone has considered introducing the Differential Evolution (DE) scheme [3] into PSO. A promising new evolutionary algorithm known as DE was recently introduced and has garnered significant attention in the research literature. Compared with other forms of evolutionary algorithms, it hardly requires any parameter tuning and is very efficient and reliable. Hybridizing PSO with DE is proposed according to the abilities of PSO in remembering knowledge and thus good solutions are retained by all the particles. On other hand, DE discards previous knowledge of the problem when the population changes. Moreover, PSO and DE both work with an initial population of solutions. Therefore, combining the searching abilities of both methods seems to be a reasonable approach that's led to our proposed hybrid

PSODE algorithm by using the DE technique to generate the PSO particles and try to increase the social search by using the global knowledge of each particle while updating the particles. The structure of this paper is organized as follows. In Section 2, a brief review of related works is presented. The details of PSO and DE are explained in Section 3 and Section 4, and then structure of the new hybrid PSODE and frameworks are shown in section 5. Finally the experimental results on some well-known benchmark functions and discussion are present in section 6.

## 2. Related Works

Hybridization has turned out to be an effective and efficient way to design high-performance optimizers, which is witnessed by the rapid evolution of diverse hybrid optimizers in the past decade. As a special and representative member in the family of hybrid optimizers, DEPSO has received much attention from researchers that are interested in optimization, problem solving, and algorithm design, therefore the researchers start to propose hybrid PSO variants to optimize many different optimizations problems such as multimodal functions [4]. [5] And [6] combined PSO and DE operators to search for global solutions of multi-modal functions, while [7] introduced stochastic local search in PSO for multi-modal function optimization. DEPSO has been further hybridized with other optimizers that led to more complicated architecture [8] and [9]. [10] proposed DEPSO algorithm for unconstrained optimization to enhance the algorithm's exploration ability by using three updating strategies for each particle (a- DE update strategy(DEUS), b- random update strategy and c- PSO update strategy where the last two ways were combined together to get Combined Updating Strategy (CUS)). Other researches proposed the hybridization of PSO with local search [11, 12], where [13] and [14] merged hybridization of PSO with local search with other global optimization techniques. However, a few of them considered introducing the DE scheme into PSO. [15] Proposed a simple hybrid version of DE and PSO, which starts with the usual DE and incorporated PSO to reach to the optimal solution. [16] developed a hybrid version consisting of Barebones PSO and DE where [17] found the candidate solution is generated either by DE or by PSO according to some fixed probability distribution. [18] Presented DEPSO-2S, a hybridization of DE and PSO to initialize the particles of the auxiliary swarms, rather than using the standard PSO to construct the main swarm. So far, all the above literature can be cluster the hybrid DEPSOs into three classes according to their basic types [19]:

1) collaboration-based DEPSO;
2) embedding-based DEPSO; and
3) assistance-based DEPSO.

These three classes are depending on the *relationship* between parent optimizers for hybridization.

## 3. Overview of Differential Evolution Algorithm

DE was proposed about the same time as PSO by Storn and Price [20] for global optimization over continuous search space. Its theoretical framework is simple and requires a relatively few control variables but performs well in convergence. For some unknown reason, DE caught on much slower than PSO but has lately been applied and shown its strengths in many application areas ([21]; [22]). DE is formidable population-based optimizers, and it was extensively used in practice [23]. It was founded, on the same structure of Genetic Algorithm and suggest by the Nelder–Mead simplex method [24]. This method can optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. Such methods are commonly known as meta-heuristics as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. DE consists of three operators a- mutation, b- crossover, and c- selection] these operators are the same to Genetic Algorithm operators. But, the mutation operation in DE is different from the mutation in GA, because in DE it is depend on the difference of different individuals, copying ideas from the Nelder–Mead simplex method. The parameters of DE can be representing as $x$, $y$, $z$, where $x$ identifies the base vector to be mutated, $y$ is the number of difference vectors used, and $z$ symbolizes the crossover scheme [24]. The classical DE variant, has mutation of the $i$th individual in the DE population $\{\mathbf{x}_i \mid i = 1, 2, \ldots, \mathrm{PS}\}$, where PS is the population size three different individuals $\mathbf{x}_{r1}$ , $\mathbf{x}_{r2}$ , and $\mathbf{x}_{r3}$ with $r1 \neq r2 \neq r3 \neq i$ will be randomly (rand) chosen from the population to generate a new vector. The new vector can be expressed as follows:

$$\mathbf{z}_i = \underbrace{x_{r1}}_{\text{base}} + \underbrace{F * (x_{r2} - x_{r3})}_{\text{Individual .difference.}} \qquad (1)$$

Where $F$ is the so-called scaling factor, which is a positive constant. A general setting for this factor is $F \in [0, 2]$. But, Storn and Price suggest $F \in [0.5, 1]$ as such a setting may result in good optimization effectiveness. Next step followed

mutation is the crossover operates on the vector $\mathbf{z}_i$ and the target vector $\mathbf{x}_i$ to generate the final vector $\mathbf{u}_i$ in the following way:

$$U_{id} = \begin{cases} z_{id} \ if \ rand\ d_i \le CR \ or \ d = rn_i \\ x_{id}, \ otherwise \end{cases} \quad (2)$$

where $x_{i,d}$ , $z_{i,d}$ , and $u_{i,d}$ are the $d$th dimensional components of the vectors $\mathbf{x}_i$ , $\mathbf{z}_i$ , and $\mathbf{u}_i$ , sequentially; CR represent the crossover probability, which is usually set to a fixed value in (0,1) or changes dynamically within (0,1); $rn_i$ is a number that is randomly chosen from the index set $\{1, 2, \ldots, D\}$ and this number used to make sure that the trial vector $\mathbf{u}_i$ is unlike the original solution $\mathbf{x}_i$.

At the end, $\mathbf{u}_i$ will be compared with $\mathbf{x}_i$ , and the better of them will be pick out to be a new member of the DE population for the next generation. This replacement scheme is *de facto* a one-to-one tournament selection.

There is other typical DE variant [20]. It is similar to the stander DE except for the mutation strategy. The new variant of DE mutation can be defined as follows:

$$z_i = \underbrace{x_{best} + F}_{\text{base}} * \underbrace{(x_{r1} - x_{r2} + x_{r3} - x_{r4})}_{\text{indiv.diff.}} \quad (3)$$

Wnere $\mathbf{x}_{\text{best}}$ is the best individual in the current population; $\mathbf{x}_{r1}$, $\mathbf{x}_{r2}$, $\mathbf{x}_{r3}$, and $\mathbf{x}_{r4}$ are four different individuals and they are randomly selected from the current population. The following two DE mutation strategies are also can be used.

DE [25].

$$z_i = \underbrace{x_i + \lambda * (x_{best} - x_i)}_{\text{base}} + \underbrace{F * (x_{r1} - x_{r2})}_{\text{indiv.diff.}} \quad (4)$$

DE [26].

$$z_i = \underbrace{(x_i + x_{better})/2 + \lambda * (x_{better} - x_i)}_{\text{base}} + \underbrace{F * (x_{r1} - x_{r2})}_{\text{indiv.diff}}$$

At this point, $\mathbf{x}_{\text{better}}$ ıs an unique that ıs select from the DE population in random way, and the fitness value of it should be better than or equal to that of $\mathbf{x}_i$ ; and $\lambda$ is a scale coefficient with $0 < \lambda < 1$, and it is usually set to the same value as $F$.

## 4. Overview of Particle Swarm Optimization (PSO)

In 1995, a paper on PSO was presented at the Congress on Evolutionary Computation [27]. This landmark paper triggered waves of publications in the last decade on various successful applications of PSO to solve many difficult optimization problems.

It is inspired by the flocking and foraging behavior of birds and fish [28]. It has highly desirable attributes such as easy to understand and implement. In addition, the main strength of PSO is its fast convergence as compared with other optimization algorithms such as, Simulated Annealing (SA) and Genetic Algorithms (GA) [29].

It is very appealing because of the simple conceptual framework and the analogy of birds flocking facilitated conceptual visualization of the search process. In PSO, a solution is represented as a particle, and the population of solutions is called a swarm of particles. Each particle has two main properties: position and velocity. Each particle moves to a new position using the velocity. Once a new position is reached, the best position of each particle and the best position of the swarm are updated as needed. The velocity of each particle is then adjusted based on the experiences of the particle.

The velocity (Vi) of each particle is updated using the following equation:

$$v_i^{t+1} = wv_i^t + c_1 * rand * (pbest_i - x_i^t) + c_2 * rand * (gbest - x_i^t)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}$$

Where

$V_i^t$ : is the velocity of particle *i* at iteration *t*,

*w:* is a weighting function,

$C_1$: is the individual coefficient,

$C_2$: is the social coefficient

Rand: is a random number between 0 and 1,

$X_i^t$: is the current position of particle i at iteration t,

$V_i^{t+1}$: is the current velocity of particle i at iteration t+1,

$pbest_i$: is the pbest of agent i at iteration t,

gbest: is the best solution so far.

The process is repeated until a stopping criterion is met. Similar to GA, the first process of PSO is initialization whereby the initial swarm of particles is generated. The concept of solution representation is also applied here in very much the same manner as GA. Each particle is initialized with a random position and velocity. Each particle is then evaluated for fitness value. Each time a fitness value is calculated, it is compared against the previous best fitness value of the particle and the previous best

fitness value of the whole swarm, and the personal best and global best positions are updated where appropriate. If a stopping criterion is not met, the velocity and position are updated to create a new swarm. The personal best and global best positions, as well as the old velocity, are used in the velocity update. As mentioned earlier, the two key operations in PSO are the update of velocity and the update of position. The velocity is updated based on three components: the old velocity (inertia or momentum term), experience of an individual particle (cognitive or self-learning term), and experience of the whole swarm (group or social learning term). Each term has a weight constant associated with it. For basic PSO algorithm, the number of required constants is three. It should be noted that PSO algorithm does not require sorting of fitness values of solutions in any process. This might be a significant computational advantage over GA, especially when the population size is large. The updates of velocity and position in PSO also only require a simple arithmetic operation of real numbers.

**5. Structure of The new Hybrid PSODE and Frameworks**

Both of PSO and DE algorithms are stochastic, and they are very useful to reach global optimum than gradient descent method. On the other hand, it's very simple for them to drop into the local optima, and the convergence accuracy is not satisfying. To develop these weaknesses and utilize the advantages, a hybridization of PSO and DE is presented to improve the performance of optimization. A hybrid particle swarm with differential evolution operator termed PSODE this algorithm is designed so as to preserve the strengths of the both algorithms. PSODE starts like the usual PSO algorithm up to the point where the particle is updated then we start the DE algorithm to update the particle by mutation it with Pbest , then make the crossover for the result of mutation and Gbest, and last step in DE is select the best particle and update our data. After these steps we return Gbest as optimal solution from running PSO algorithm. **Figure** 1 and 2 illustrate the pseudo code and flowchart of PSODE. As shown in these figures, the positions of particles are updated by using DE algorithm and each particle is evaluated based on its fitness value. Finally, the best solution is returned by the algorithm.

```
For each particle
        Initialize particle with random position and velocity vectors
Repeat until meeting end criterion
     For each particle
        Begin
          Calculate fitness value
           If fitness value is better than the best fitness value ( Pbest) in history
            set current value as the new Pbest
        End
For each particle
  Begin
   Calculate particle velocity
    Start DE
      Mutation( particle , Pbest )
      Crossover (mutation , Gbest)
       Select the best particle and update our data
    End DE
  End
Stop: return Gbest as optimal solution
```

**Figure 1.** PSODE pseudo code.

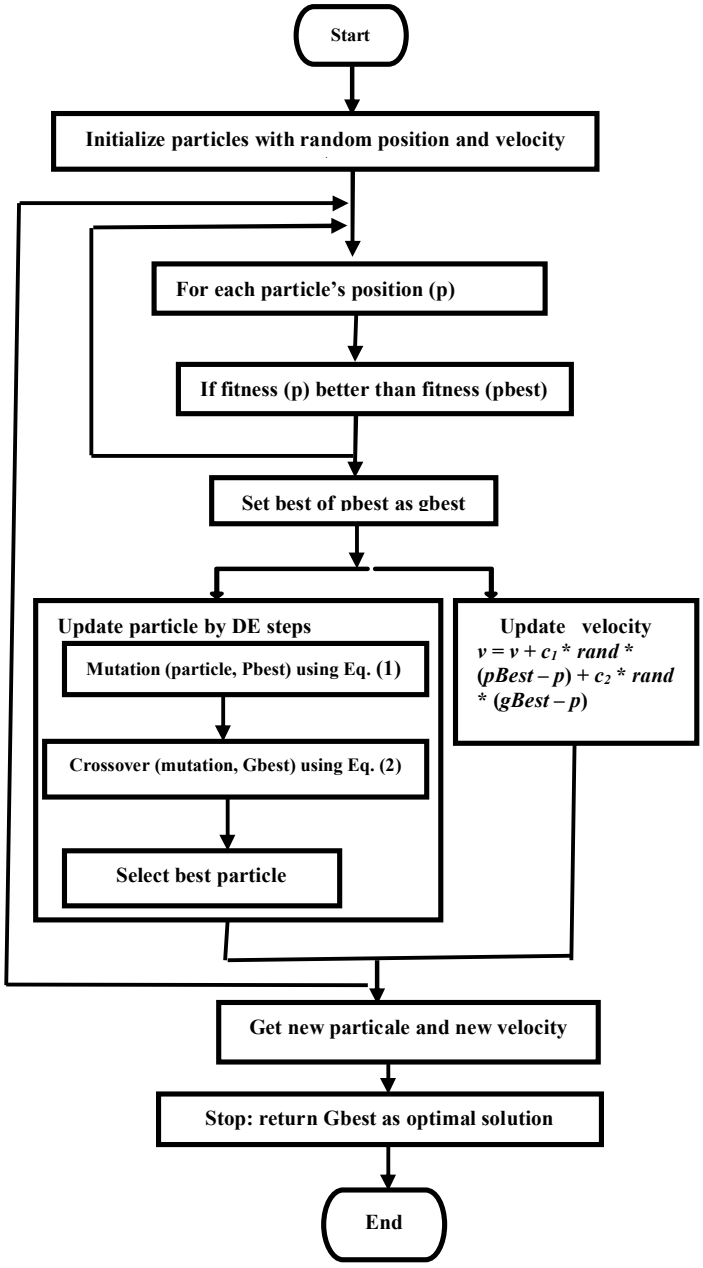


**Figure 2.** the Flowchart of the proposed hybrid PSODE algorithm.

## 6. Experimental results and discussion

To evaluate the performance of PSODE, 8 minimization and maximization benchmark functions are selected including 4 real life problems benchmark as detailed in Section 6.1.

In **Table 1** Range and n are the feasible bound and the dimension of function respectively.

**Table 1.**

| Test function | Dimension (n) | Range |
|---|---|---|

| | | |
|---|---|---|
| F1 | 3 | x1[10,55], x2[1.1, 2] , x3[10, 40] |
| F2 | 2 | x1 [17.5, 40] , x2[300, 600] |
| F3 | 4 | [12, 60] |
| F4 | 3 | x1[0.02, 0.8], x2[10, 40], x3[3000, 20000] |
| F5 | 30 | [-100, 100] |
| F6 | 30 | [-5.12, 5.12] |
| F7 | 30 | [-32.768, 32.768] |
| F8 | 30 | [-30,30] |

**6.1 Benchmark functions:**

1- Gas transmission design (F1)

$$Min\, f(x) = \left\{ \begin{matrix} 8.61 * 10^5 x_1^{\frac{1}{2}} x_2 x_3^{-2/3} (x_2^3 - 1)^{-1/2} + 3.69 * \\ 10^4 x_3 + 7.72 * 10^8 x_1^{-1} x_2^{0.219} - 765.43 * 10^8 x_1 \end{matrix} \right\}$$

2- Optimal capacity of gas production facilities (F2)

$$Min\, f(x) = \left\{ \begin{matrix} 61.8 + 5.72x_1 + 0.2623 \left[(40 - x_1) ln\left(\frac{x_2}{200}\right)\right]^{-0.85} + \\ 0.087(40 - x_1) ln\left(\frac{x_2}{200}\right) + 700.23 x_2^{-0.75} \end{matrix} \right\}$$

3- Design of a gear train (F3)

$$Min\, f(x) = \left\{\frac{1}{6.931} - \frac{T_d T_b}{T_a T_f}\right\}^2 = \left\{\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right\}^2$$

4- Optimal thermo hydraulic performance of an artificially roughened air heater (F4)

$$Max\, L = 2.51\, ln\, e^+ + 5.5 - 0.1 R_M - G_H$$

Where:

$$R_M = 0.95 x_2^{0.53} \quad G_H = 4.5 (e^+)^{0.28} (0.7)^{0.57} \quad e^+ = x_1 x_3 \left({}^{F}/_{2}\right)^{1/2}$$

$$F = (f_s + f_r)/2, \quad f_s = 0.079 x_3^{-0.25}$$

$$f_r = 2(0.95 x_3^{0.53} + 2.5 ln\left({}^{1}/_{2x_1}\right)^2 - 3.75)^{1/2}$$

5- Sp Shpere(n variables) (F5)

$$SP_n(x) = \sum_{i=1}^{n} x_i^2$$

6- Rastrigin's function (F6)

$$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$$

7- Ackley's function (F7)

$$f(x) = -a * \exp\left(-b * \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(cx_i)\right) + a + \exp(1)$$

Where: a= 20, b = 0.2 , c = 2*π

8- Rn Rosenbrock (F8)

$$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 + x_i)^2]$$

**6.2 Results and Discussions**

The proposed algorithm is compared with two algorithms which proposed by [18] namely DEPSO-

2S, PSO-2S as well as with the classical PSO and DE where the parameters of PSO were set as (Number of particles and Number of iteration were set during the run, C1, C2 = 1.19, R1, R2 = 1), for DE the parameters were set as (Population size and Number of iteration and Problem dimension were set during the run, Maximum population size=100, Cr =0.5, F= 0.7). The averages of the best value for 20 run times and the maximum number of the function evaluations (*Nb. evals*) depends on each problem is illustrated in Table 2 and Table 3.

**Table 2**. Comparative the Proposed Method with a Different Four Algorithms Based on a Real Life Problem Benchmark Functions

| | Nb. evals | DE | PSO | PSO-2S | DEPSO-2S | PSODE |
|---|---|---|---|---|---|---|
| **F1** | **24000** | **1.6929 e+006** | **1.6894 e+006** | **7.43233e+006** | **2.96438e+006** | **1.6888 e+006** |
| **F2** | **16000** | **1.6987 e+002** | **1.6988 e+002** | **1.69844e+002** | **1.69844e+002** | **1.6983 e+002** |
| **F3** | **32000** | **0.9922 e-008** | **0.9623 e-008** | **1.40108e-010** | **1.39732e-010** | **0.9706 e-010** |
| **F4** | **24000** | **4.2053 e-005** | **4.1986 e-005** | **2.31987e-006** | **3.17128e-005** | **3.84631 e-005** |

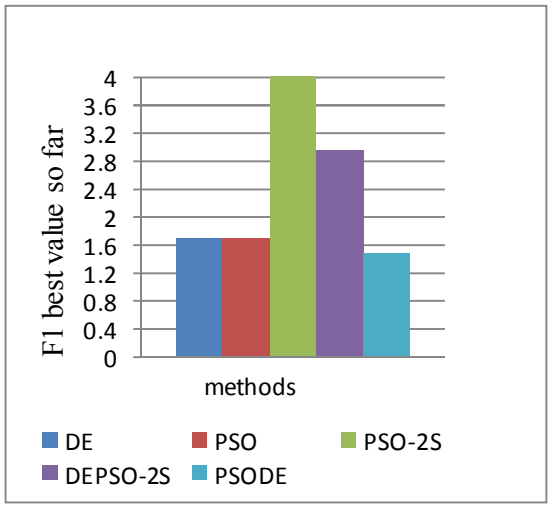


**Figure 1.** Performance Comparisons of DE, PSO, PSO-2S, DEPSO-2S, PSODE (proposed method) for function F1

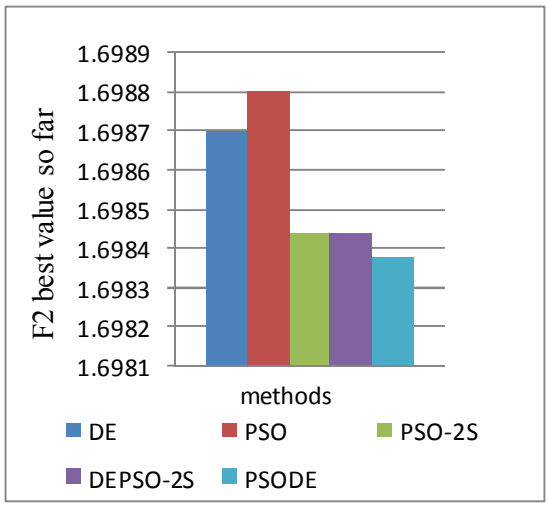


**Figure 2.** Performance Comparisons of DE, PSO, PSO-2S, DEPSO-2S, PSODE (proposed method) for function F2

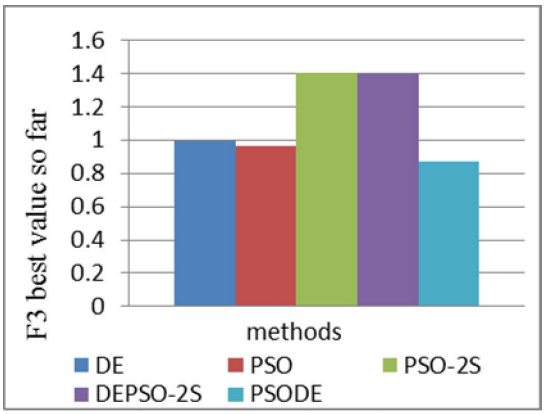


**Figure 3**. Performance Comparisons of DE, PSO, PSO-2S, DEPSO-2S, PSODE (proposed method) for function F3

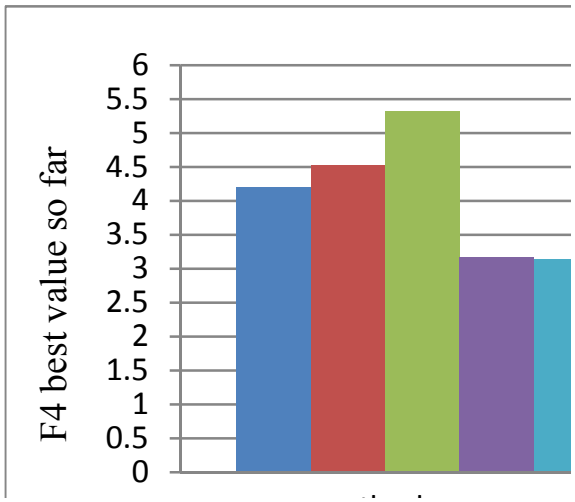


**Figure 4.** Performance Comparisons of DE, PSO, PSO-2S, DEPSO-2S, PSODE (proposed method) for function F4

From the experiments, we can notice that PSODE obtains the best results on most of the problems used. Thus, this algorithm leads to a remarkable improvement compared to the previous classical PSO, DE and for PSO-2S; DEPSO-2S. PSODE outperforms the other tested algorithms.

**Table3.** Comparison of classical PSO and DE with proposed PSODE on Stander test function benchmark

| | | PSO | DE | PSODE |
|---|---|---|---|---|
| F5 | Best | $2.131\times10^{-12}$ | $1.4364\times10^{-12}$ | $2.2962\times10^{-24}$ |
| | Worst | $4.2322\times10^{-2}$ | $8.7664\times10^{-7}$ | $1.1863\times10^{-11}$ |
| | Std | $6.6875\times10^{-3}$ | $1.3406\times10^{-7}$ | $1.9369\times10^{-12}$ |
| F6 | Best | 32.8287 | 21.2728 | 13.9294 |
| | Worst | 127.3341 | 175.3698 | 65.6672 |
| | Std | 19.0037 | 35.8085 | 9.2816 |
| F7 | Best | $7.3543\times10^{-6}$ | $8.4260\times10^{-8}$ | $7.4252\times10^{-13}$ |
| | Worst | 5.9074 | 2.3162 | 1.5017 |
| | Std | 1.2183 | 0.6521 | 0.3366 |
| F8 | Best | 2.0718 | 18.3952 | 0.0248 |
| | Worst | 485.0228 | 577.7486 | 225.5721 |
| | Std | 76.5658 | 91.5654 | 41.6379 |

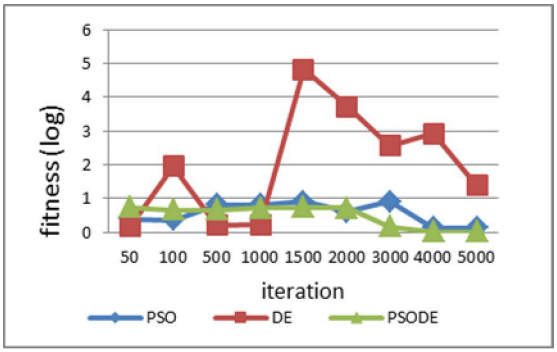


**Figure 5.** The evolution curve of Sphere function

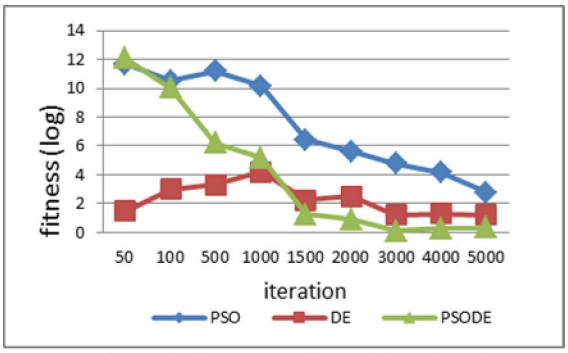


**Figure 6.** the evolution curve of Rastrigin's function

In Figure 5 we got the best solution for PSODE algorithm in iteration 4000 and 5000 with domain (n=30) for Sp Shpere function, where in Figure 6 and also Figure 7 we got the minimum result for the proposed algorithm PSODE in iteration 3000 and in iteration 4000 with domain (n=30) for the Rastrigin's function and for Ackley's function . while we got the best value in iteration 1500 for Rn Rosenbrock with domain (n=30)So far the performance was good in all of the previous functions.

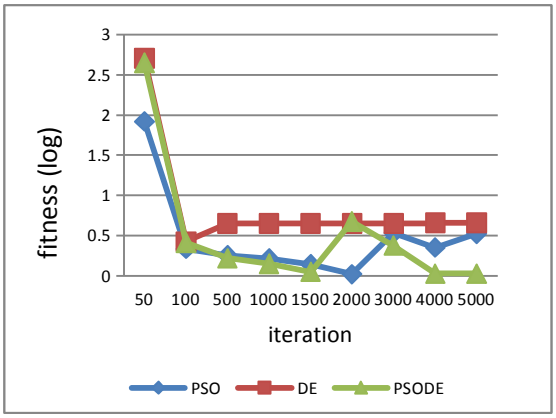


**Figure 7.** the evolution curve of Ackley's function

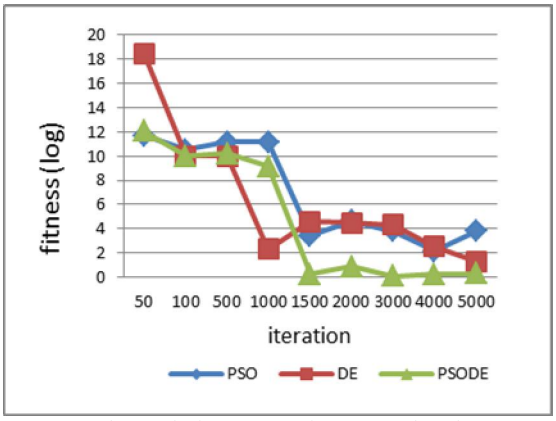


**Figure 8.** the evolution curve of Rn Rosenbrock

## 7. Conclusions

A new method named PSO-DE is introduced in this paper, which improves the performance of the particle swarm optimization by incorporating differential evolution. Our aim is to use the proposed algorithm to update PSO particles by using

differential evolution selection and mutation and crossover that led to enhance the searching abilities of PSO. As a result, the proposed algorithm has the automatic balance ability between exploration and exploitation. The approach obtains competitive results on 8 well-known benchmark functions including 4 real-life well known problems. Computational results show that the proposed hybrid approach gives more accurate results compared to other evolutionary algorithms.

**References**


1. J. Kennedy, R.C. Eberhart, Y. Shi, Swarm Intelligence, Morgan Kaufmann Publisher, San Francisco, 2001.
2. Q.J. Guo, H.B. Yu, A.D. Xu," A hybrid PSO-GD based intelligent method for machine diagnosis", Digital Signal Processing vol.16, No 4, pp.402_418, 2006.
3. R. Storn and K. Price, "Differential evolution—A simple and efficient heuristics for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
4. Wenlong Fu, Mark Johnston, and Mengjie Zhang, Hybrid Particle Swarm Optimization Algorithms Based on Differential Evolution and Local Search, AI 2010, pp. 313–322, 2010.
5. Zhang,W., Xie, X.: DEPSO: hybrid particle swarm with differential evolution operator. In: IEEE International Conference on Systems, Man & Cybernetics (SMCC), Washington DC, USA, pp. 3816–3821, 2003.
6. Xin, B., Chen, J., Peng, Z., Pan, F.: "An adaptive hybrid optimizer based on particle swarm and differential evolution for global optimization". Science China Information Sciences vol.53, No 5, pp.980 –989, 2010.
7. R.Akbari & K. Ziarati 2008. Combination of Particles Swarm Optimization and Stochastic Local Search for Multimodal Function Optimization, 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Applications, Pages 388-392, Wuhan, China 2008.
8. C. Potter, G. K. Venayagamoorthy, and K. Kosbar, "RNN based MIMO channel prediction," *Signal Process.*, vol. 90, no. 2, pp. 440–450, 2010.
9. K. Vaisakh, P. Praveena, and S. Rama Mohana Rao, "DEPSO and bacterial foraging optimization based dynamic economic dispatch with nonsmooth fuel cost functions," in *Proc. World Cong. Natur. Bio. Inspir. Comput.*, Coimbatore, India, pp. 152–157, 2009.
10. Changsheng Zhanga, Jiaxu Ning, Shuai Lu, Dantong Ouyang, Tienan Ding, "A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization", Elsevier B.V. Operations Research Letters ,vol.37, pp.117_122, 2009.
11. S.K. Fan, E. Zahara, "A hybrid simplex search and particle swarm optimization for unconstrained optimization", European Journal of Operational Research vol.181, pp.527_548, 2007.
12. Q.J. Guo, H.B. Yu, A.D. Xu," A hybrid PSO-GD based intelligent method for machine diagnosis", Digital Signal Processing vol.16, No 4, pp.402_418, 2006.
13. Y.T. Kao, E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions", Applied Soft Computing vol.8, No 2, pp.849_857, 2008.
14. P.S. Shelokar, Patrick Siarry, V.K. Jayaraman, B.D. Kulkarni, Particle swarm and ant colony algorithms hybridized for improved continuous optimization, Applied Mathematics and Computation vol.188, No.1, pp.129_142, 2007.
15. Millie Pant, Radha Thangaraj, Crina Grosan, Ajith Abraham, "Hybrid Differential Evolution – Particle Swarm Optimization Algorithm for Solving Global Optimization Problems", IEEE Digital Information Management, ICDIM . Third International Conference , pp. 18 – 24, 2008.
16. Omran, Mohd. G.H., Engelbrecht, A. P., Salman, Ayed, "Differential Evolution based Particle Swarm Optimization", IEEE Swarm Intelligence Symposium (SIS 2007), pp. 112 – 119, 2007.
17. Zhi Feng Hao, G. H. Guo, and H. Huang, "A particle swarm optimization algorithm with differential evolution," in *Proc. 6th Int. Conf. Mach. Learn. Cybern.*, Hong Kong, China, , pp. 1031–1035, 2007.
18. Abbas El Dor, Maurice Clerc, and Patrick Siarry," Hybridization of Differential Evolutionand Particle Swarm Optimization in a New Algorithm: DEPSO-2S", Springer-Verlag Berlin Heidelberg, pp. 57–65, 2012.
19. Bin Xin, Jie Chen, Juan Zhang, Hao Fang, and Zhi-Hong Peng," Hybridizing Differential Evolution and Particle Swarm Optimization to Design Powerful Optimizers: A Review and Taxonomy", IEEE, 2011.
20. R. Storn and K. Price, "Differential evolution—A simple and efficient heuristics for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
21. Godfrey, O. and Donald, D. "Scheduling flow shops using differential evolution algorithm",

*European Journal of Operational Research*, vol.171, no.2, pp.674-692, 2006.
22. Qian, B., Wang, L., Huang, D.-X., and Wang, W. "Scheduling multi-objective job shops using a memetic algorithm based on differential evolution", The International Journal of Advanced Manufacturing Technology, vol.35, pp.1014-1027, 2008.
23. S. Das and P. N. Suganthan, "Differential evolution: A survey of the state of the art," *IEEE Tran. Evol. Comput.*, vol. 5, no. 1, pp. 4–31, 2011.
24. K. Price, R. M. Storn, and J. A. Lampinen, "*Differential Evolution: A Practical Approach to Global Optimization* (Natural Computing Series)", 1st ed. New York: Springer-Verlag, 2005.
25. A.K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr.2009.
26. Zhi Feng Hao, G. H. Guo, and H. Huang, "A particle swarm optimization algorithm with differential evolution," in *Proc. 6th Int. Conf. Mach. Learn. Cybern.*, Hong Kong, China, , pp. 1031–1035, 2007.
27. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks IV, pp. 1942–1948 ,1995.
28. Brownlee, J. (2011). *Clever Algorithms*: Jason Brownlee.
29. Abraham, A., Guo, H. and Liu, H. "Swarm Intelligence: Foundations, Perspectives and Applications". In N. Nedjah & L. Mourelle (Eds.), *Swarm Intelligent Systems*, Springer Berlin / Heidelberg, 26, 3-25, 2006.

6/12/2021