ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 10(18), DOI: 10.17485/ijst/2017/v10i18/104743, May 2017* ISSN (Online) : 0974-5645

Enhanced DE with Weighted Base Vector for Unconstrained Global Optimization

Pravesh Kumar[1*], Millie Pant[2], Musrrat Ali[3] and H. P. Singh[4]

[1]Department of Mathematics, Jaypee Institute of Information Technology, Noida – 201301, Uttar Pradesh, India; praveshtomariitr@gmail.com
[2]Department of Applied Science and Engineering, Indian Institute of Technology Roorkee – 247667, Uttarakhand, India; millidma@gmail.com
[3]School of Technology, Glocal University, Saharanpur – 2471001, Uttar Pradesh, India; musrrat@theglocaluniversity.in
[4]Cluster Innovation Centre, University of Delhi, Delhi –110007, India; harendracicdu@gmail.com

Abstract

Objectives: Differential Evolution (DE) algorithm has came out as a robust, effective and well-organized computational technique for solving global optimization problems. However, similar to other evolutionary algorithms of the same genre, DE has some inherent drawbacks like slow/ premature convergence, stagnation of population etc. due to its probabilistic nature. This paper aims to decrease the drawbacks and hence enhance the working of DE algorithm in term of convergence speed and accuracy of result. **Method:** This paper presents two improved versions of DE named Differential evolution with weighted base vector (DEwB-1) and DEwB-2 which adapts novel mutation scheme and self adaptive approach to control DE parameters. **Findings:** The corresponding DE versions are tested on 13 standard unconstrained problems as suggested in various literatures and a real life molecular potential energy problem. The numerical and statistical results expose that the proposed modifications assist in improving the performance of basic DE algorithm. **Application:** The variants can apply on more complex and constrained optimization problems.

Keywords: Differential Evolution, Global Optimization, Molecular Potential Energy Problem, Mutation, Weighted Base Vector

1. Introduction

Evolutionary Algorithms (EAs) are population based stochastic search techniques famous for their capacity to handle complicated non-linear optimization problems. The primary advantage of EAs over other numerical methods is that they just require the objective function values, while mathematical properties such as differentiability and continuity are not necessarily needed. Some popular techniques proposed under the evolutionary computation field are genetic algorithms, evolutionary algorithms, genetic programming, evolutionary programming, and evolution strategies. Differential Evolution (DE)[1] has appeared as easiest and well-organized algorithm to solve optimization problems. Some of its numerous attractive qualities are simple structure, robust and good convergence speed compare of other evolutionary algorithms. It has been effectively implemented to a miscellaneous domain of science and engineering, such as pattern recognition, image processing, power engineering, engineering design, and many other problems arise in engineering and science field[2-8] and has been verified to be better than numerous of its counterpart.It has already been proved in evolutionary computation literature that DE provides better accuracy and high convergence speed than other evolutionary algorithm. However, convergence speed of DE does not meet up the prospects some time, mainly in case of high dimensional or highly multimodal problems[9].

A number of cases are exists in literature to increase the performance of DE. Some of them are as; In 2003, a Trigonometric Mutation Operator (TMO) is

* *Author for correspondence*

recommended and named it as Trigonometric DE (TDE)[10]. Some other variants are (DEahcSPX)[11], Simplex DE[12], clustering-based DE Learning Enhanced DE (LeDE)[13], DE with orthogonal crossover[14], DE/rand-to-best-best/2[15]. These DE variants contain some additional component which uses DE as an evolutionary structure assisted by extra algorithmic components. The other category contains those variants which makes an important modification within the structure of DE. Some of known example of these are, Fuzzy Adaptive DE (FADE)[16], Self adaptive control parameter based DE (jDE)[17], Random localization based DE(DERL)[18], DE with preferential crossover[17], Opposition-based DE (ODE)[18], Global and Local neighbourhood based DE (DEGL)[19], Self-adaptive DE (SaDE)[20], Adaptive DE with optional external archive (JADE)[21], DE with chaotic local search[22], DE with Ensemble of mutation strategies and control Parameters (EPSDE)[23], Modified Randomomized Localization based DE (MRLDE)[24], Dynamic parameter selection for DE[25], Cultivated DE[26], Information Utilization Selection Based DE (IUDE)[27]and so on. A complete analysis on DE variants can be found also in[28-30].

During some last years, a lot of research work on control parameter for DE is proposed. The major control parameters in DE algorithm are scaling factor *F* and crossover rate *Cr*. Storn and Price suggested that a well primary choice of *F* can be 0.5 and 0.9 for *Cr*. Instead of manual fixing of parameters, some researchers also suggested adaptive/self-adaptive approaches where the control parameters are distorted vigorously based on some response of the search region. A few of the research done in the improvement of adaptive/self-adaptive control parameters approaches can be found in[32-35].

Two modified variants of DE named as DEwB-1 and DEwB-2 are presented in the paper. Both variants are based on mutation schemes in which the base vector is taken as the weighted mean of three vectors randomly chosen from the population to perform mutation operation. A new adaptive strategy is also proposed to select the scaling factor and crossover constant which are the key control parameters of DE algorithm.

2. Brief Description of DE

Like other evolutionary algorithm, DE also works in two phases: initialization and evolution. In the first phase, a population is generated randomly for DE when no preliminary knowledge is known about the problem. In evolution phase, all the individuals go through mutation, crossover and selection process repeatedly until the termination criterion is met.

The working of DE is as follows:

2.1 Initialization

As we know that, DE is a population based evolutionary algorithm, so we start by generating a uniformly distributed population of individuals. After generating the population we evaluate the fitness value of all individuals.

Let the population of size *NP* at any generation G is denote by $P^G = \{X_i^G \mid i = 1,2,\cdots,NP\}$ where each individual $X_i^G = \{x_{1,i}^G, x_{2,i}^G, \cdots, x_{D,i}^G\}$ is a *D*-dimensional vector. Each vector of the starting population (at *G*=0) can be generate as given in equation-1.

$$X_i^G = X_{low} + (X_{upp} - X_{low}) \times rand^{(0,1)} \quad (1)$$

Where X_{upp} and X_{low} are initial higher and lower bounds for space and $rand^{(0,1)}$ are uniformly distributed random numbers having value between 0 and 1.

2.2 Mutation

In the mutation phase we generate a perturbed or mutant vector V_i^G corresponding to each individual X_i^G. For this purpose first we select 3 mutually different individuals say X_{r1}^G, X_{r2}^G, and X_{r3}^G, from the population randomly and obtain V_i^G by equation-2.

$$V_i^G = X_{r1}^G + F(X_{r2}^G - X_{r3}^G) \quad (2)$$

where *F* is a real and constant factor called scaling factor and use to control the magnification of the difference vector $(X_{r2}^G - X_{r3}^G)$. As suggested by Storn and Price[1] the value of F is should be taken any value from 0 to 1

2.3 Crossover

The next phase of DE algorithm is crossover operation where we generate a *trial* vector $U_i^G = (u_{1,i}^G, u_{2,i}^G \ldots, u_{D,i}^G)$ by combine the target vector $X_i^G = (x_{1,i}^G, x_{2,i}^G \ldots, x_{D,i}^G)$ and perturbed vector $V_i^G = (v_{1,i}^G, v_{2,i}^G \ldots, v_{D,i}^G)$. The components $(u_{j,i}^G)$ of U_i^G are generated as follows;

$$u_{j,i}^G = \begin{cases} v_{j,i}^G & \text{when} \quad rand_j^{(0,1)} \le Cr \vee j = j_0 \\ x_{j,i}^G & \text{otherwise} \end{cases} \quad (3)$$

where $j \in \{1,\ldots, D\}$, j_0 is a random parameter index, elected just the once for each i. C_r is another important control parameter which is also known as crossover factor or crossover probability and generally having value as $C_r \in [0, 1]$.

2.4 Selection

The last operation of DE algorithm is selection operation. In this operation target and trial vectors compare with each other by their fitness value and retain in next generation with minimum fitness value. The selection process of vector for next generation is:

$$X_i^{G+1} = \begin{cases} U_i^G & f \quad f(U_i^G) \le f(X_i^G) \\ X_i^G & otherwise \end{cases} \tag{4}$$

3. DE with Weighted base Vector (DEwB)

In this Section, the proposed variants *DEwB-1* and *DEwB-2* are described. The mutation strategies and self adaptive control parameter strategies are given below:

3.1 Proposed Mutation Strategies

In the proposed algorithms (DEwB-1 and DEwB-2), the base vector is selected either as a random vector or as a weighted mean of three selected vectors during the mutation operation. The idea of taking base vector as a weighted mean of selected individuals was first used by[34] and was called improved donor formulation for DE. In our study, an enhanced variant of DE is proposed by modifying the original idea of[34]. In the proposed variant, first a probability (P_d) is fixed and a uniformly distributed random number (R) is created having value between 0 and 1. Now select the base vector by weighted mean for smaller value of R than P_d, otherwise select it randomly (as per equation-2)

If X_w is the base vector obtained by the new strategy, then it may be defined as:

DEwB-1: $$X_w^G = (\mu_1 X_{r_1}^G + \mu_2 X_{r_2}^G + \mu_3 X_{r_3}^G) \tag{5}$$

DEwB-2: $$X_w^G = (\mu_1 X_{best}^G + \mu_2 X_{r_1}^G + \mu_3 X_{r_2}^G) \tag{6}$$

Here μ_1, μ_2, μ_3 are uniformly distributed random number between (0, 1) and should satisfy the condition; $\sum_i \mu_i = 1$ which means that there should be a convex combination between X_{r1}^G, X_{r2}^G and X_{r3}^G.

The mutation scheme may be represented as DE/donor/1 and the corresponding proposed variant is called DE with weighted base vector (DEwB).

From equation-5 and 6, it can be seen that the only difference between basic DE and DEwB is in the selection of base vector. In DEwB, we take base vector as the convex combination of three selected vectors X_{r1}^G (X_{best}^G for DEwB-2) X_{r2}^G and X_{r3}^G. From the condition it can also be seen that if we take $\mu_1 = 1$ then $\mu_2 = \mu_3 = 0$, then DEwB-1 and DEwB-2 will be similar to DE/rand/1 and DE/best/1 variants respectively. A graphical explanation of DEwB-1 is given in Figure 1.

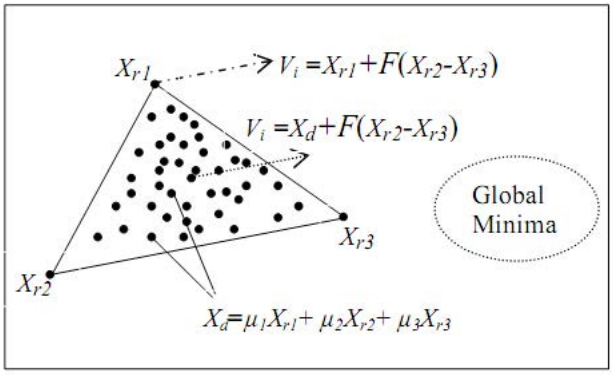


Figure 1. Graphical explanation of DEwB-1 mutation process.

By convex combination of vectors we have a benefit that each of the newly weighted vectors will remain inside the boundary of triangular region for which X_{r1}^G, X_2^G and X_{r3}^G are vertices. Out of these weighted vectors, only one vector will produce randomly for i^{th} target vector. From the Figure 1, it can be noticed that by using donor mutation scheme, the mutant vector will possibly be at a smaller distance from the global minima rather than basic mutation scheme. Hence we may get a fast convergence speed by using donor mutation scheme. But a problem may also occur with this scheme, if we use donor mutation scheme continuously then sometime the algorithm may behave like a greedy algorithm which may cause lack of diversity in the algorithm. Therefore, to minimize these risks, we fixed a probability, so that it can have a choice to select the base vector either as a weighted or random vector. By fixing the probability we can also acquire the advantage of both DE/rand/1 and DE/donor/1 schemes and increase the convergence speed in a systematic manner.

3.2 Control Parameter Setting for DEwB Algorithms

In the basic DE algorithm, the values of *F* and *Cr* are kept constant throughout the search process which may not work efficiently for every problem. The manual tuning of these parameters is a complicated task in itself due to the compound interactions, even if one carries out preliminary experimentation, the optimal parameter settings may never be found. Hence, self-adaptive control parameters can play a major role in such a situation. Thus, changing the control parameters values in consecutive iterations make available more randomness to the algorithm which consecutively may assist in getting better the working of algorithm in terms of exploitation and exploration.

In our study a new self adaptive control parameter approach is proposed to change the control parameter *F* and *Cr* during the run. For each X_i^G, control parameter F_i^G and $\mathcal{C}_i^G$ are defined in equation-7.

$$F_i^G = \begin{cases} F_l + (F_u - F_l) * \text{rand}_1 & \text{if rand}_2 < \text{Pf} \\ (F_l + F_u)/2 & \text{otherwise} \end{cases}$$
$$\text{Cr}_i^G = \begin{cases} \text{Cr}_u - (\text{Cr}_l * \text{rand}_3) & \text{if rand}_4 < \text{Pc} \\ (\text{Cr}_l + \text{Cr}_u)/2 & \text{otherwise} \end{cases} \quad (7)$$

Here $rand_1$, $rand_2$, $rand_3$, and $rand_4$,are uniformly distributed random numbers between 0 and 1, *Pf* and *Pc* represents the probability to adjust *F* and *Cr* respectively.

Cr_l and F_l, are lower bound, Cr_u and F_u are upper bound for *Cr* and *F* respectively. In the paper these constants are taken as have taken *Pf*=*Pc*=0.5, $F_l = Cr_l$=0.1 and $F_u = Cr_u$ =0.9. So that *F* and *Cr* always having a value between 0.1 and 0.9, also if the condition is not satisfied then value of *F* and *Cr* will be 0.5 which have been considered as the most suitable control parameter values by many researchers.

The main contribution of our proposed variants is that the rule for self adaptive control parameters is very simple and there is no need to guess particular values of *F* and *Cr*.

The pseudo-code of proposed variants is given in Table 1.

Table 1. Pseudo-code of proposed DEwB

Initially at *G*=0, generate uniformly distributed random population $P^G=\{X_1^G, X_2^G..., X_{NP}^G\}$ using Equation-1
calculate $f(X_i^G)$
While (Until reach to stopping condition)
{
For i=1:*NP*
{
Generate value of *F* and *Cr* as given in equation-7
Select three vectors, $X_{r1}^G X_{r2}^G$ and X_{r3}^G from P^G which are mutually
Different and also different from X_i^G.
If (*rand*<*Pr*) /* *rand* =any uniform random number between 0& 1*/
{
Generate mutant vector as defined in equation-5 (for *DEwB-1*) or by Equation-6 (for *DEwB-2*)
}
Else
{
Generate mutant vector as defined in equation-2
}
Generate trial vector U_i^G as defined in equation-3
Calculate $f(U_i^G)$
Perform selection operation to select fittest vector from X_i^G and U_i^G by using Equation-4
}* end *for* loop
Update population $P^{G+1}=\{X_1^{G+1}, X_2^{G+1}..., X_{NP}^{G+1}\}$
} /* end while loop*/
End

4. Application of DEwB-1 and DEwB-2

4.1 Benchmark Problems

General benchmark problems are selected from[13,23] for experiments. This test bed although thin, forms a launch start pad to authenticate the capability of an optimization algorithm.

4.2 Molecular Potential Energy Problem

A large number of real life problems from science and engineering may be designed as global optimization problem with variables in continuous region. In the domain of chemistry, a problem of intense current interest is to discover the lowest-energy structure of a molecule. This structure is very important because it read out many of the properties of the molecule. It seems likely that the lowest-energy structure is related to the global minimum of the molecular potential energy function[35]. The number of local minimisers that increase exponentially with the size of molecule is the main difficulty in this problem. As a result the molecular energy minimization problem is exceptionally challenging, unsolved problems in molecular biophysics and is paid much attention by chemists, physicists as well as the researchers from the field of computer science and mathematics also. Minimizing the potential energy problem of a molecule has been addressed by several researchers from time to time[35-37]. The mathematical model of this problem is highly multimodal as well as complicated in nature.

For a 3-diemnsional space, a molecule having a linear sequence of n-beads centred at y_1, y_2,...y_n. Let $L_{i,i+1}$is the bond length or simply Euclidean distance between two successive beads $y_i$and y_{i+1} and let $\psi_{,i+1}$ be the bond angle for every three successive beadsy_i, y_{i+1} andy_{i+2}, corresponding to the line having the first two with the relative location of the third bead. In the same way, let $\tau_{i,i+3}$ be the torsion angle for every four successive beads,y_i, y_{i+1},y_{i+2}, y_{i+3}between the normal's through the planes formed by the beads y_i, y_{i+1},y_{i+2} andy_{i+1}, y_{i+2},y_{i+3}.

Hence the potentials force field corresponding to bond lengths $L_{i,i+1}$, bond angles $\psi_{i,i+1}$ and torsion angles $\tau_{i,i+3}$ may be define respectively as;

$$F_1=\sum_{(i,j)\in M_1} a_{i,j}(L_{i,j}-L^0_{i,j})^2$$
$$F_2=\sum_{(i,j)\in M_2} b_{i,j}(\psi_{i,j}-\psi^0_{i,j})^2 \quad (8)$$
$$F_3=\sum_{(i,j)\in M3} c_{i,j}(1+\cos(3\tau_{i,j}-\tau^0_{i,j}))$$

where $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ are bond stretching force, angle bending force and torsion force constant respectively. The constants $\psi^0_{i,j}$ and $L^0_{i,j}$ correspond to the selected bond angle and bond length respectively. $\tau^0_{i,j}$ represent the phase angle and describe the minima location. L_j, $j=1,2,3$ represents the set of couple of atoms removed by j covalent bonds.

Furthermore, a potential $F_4$to distinguish the 2-body connections between eachcouple of beads split by more than 2 covalent bonds along the chainis defined as:

$$F_4=\sum_{(i,j)\in M_3}\left(\frac{(-1)^i}{l_{i,j}}\right) \quad (9)$$

In order to lead the optimal spatial position of the beads, total molecular potential energy $F=\sum_{i=1}^{4}F_i$ need to be minimize. Taking the parameters as defined in[35] potential energy function can be modelled as:

$$\text{Min F}=\sum_{i=1}^{n-3}(1+\cos(3\tau_{i,i+3}))+\sum_{i=1}^{n-3}\left(\frac{(-1)^i}{\sqrt{10.60099896-4.141720682\cos(\tau_{i,i+3})}}\right) \quad (10)$$

Hence the problem is to minimize F by evaluating optimal value of $\tau_{i,i+3}$ for $i=1,2,..,n$. By restricting of $\tau_{i,i+3}$; $0\le\tau_{i,i+3}\le 5$ the existence of only one global minimum is guaranteed[35].

From equation-10, it can be noticed that the given problem is a non-convex optimization problem and having numerous local minimiser even for small value of n. As defined in equation-10, the number of local minimiser of the function is 2^{n-3}. These local minimisers are corresponding to a met a stable state of the molecule which is not truly stationary but is about stationary state.

5. Experimental Settings

5.1 Parameter Settings

In order to conduct a fair judgment, all the algorithms are implemented under similar experimental settings. The value of F and Cr are taken as 0.5 and 0.9 respectively for basic DE[23] and also for other modified variants of DE which are used for comparison. Other parameter settings are given as below[13,20,23].

Population size NP=100.

For all benchmark problems, the dimension is taken as, D=30.

In case of potential energy problem, the dimension is taken as, D=10, 15, 20 and 25.

Probability factor *Pr, Pc* and *Pf* are fixed at 0.5.

Value to reach (ε) is fixed as 10^{-08} except for function F_7 where ε is fixed as 10^{-02}.

Maximum NFE are fixed as 500000 for each functions.

Software: Dev C++.

5.2 Comparison Criteria

It used five different comparison criteria which are mentioned:

5.2.1 Number of Function Evaluation (NFE)

NFE means total number of points on which fitness value has been calculated to reach $\left|f_{global} - f_{optimal}\right| \leq \varepsilon$, where ε any fixed value is and called Value To Reach (VTR). We record NFE in each runs before reach max-NFE and then find average NFE of total runs.

5.2.2 Error and Standard Deviation

In each run, we record average error which is absolute difference between global value and optimum value $|f_{global} - f_{optimal}|$ on predefined maximum NFEs. The average and standard deviation of the fitness values are also calculated.

5.2.2.1 Success Rate

Success rate is defined as: $\text{SR} = \frac{\text{Succesful Runs}}{\text{Total Runs}} * 100$

5.2.2.2 Acceleration Rate

To compare the convergence speeds between the algorithms, acceleration rate is defined as:

$$\text{AR} = \frac{\text{NFE}_{\text{others}} - \text{NFE}_{\text{DEwB}}}{\text{NFE}_{\text{others}}} \%$$

5.2.2.3 Convergence Graph

The convergence graphs represent the mean fitness performance in different generation in the respective experiments.

6. Numerical Result for Benchmark Problems

The proposed DEwB-1 and DEwB-2 algorithms are investigated by comparing it with DE algorithm and some of its enhanced variants such as TDE, LeDE, jDE, ODE, SADE, JADE and DERL, Every result is taken as average of 50 independent runs so that the effect of the stochastic nature of the algorithms can be minimized.

6.1 Comparison of DEwB-1 and DEwB-2 with DE, TDE and DERL

The simulated results and comparison of DEwB-1 and DEwB-2 with DE, TDE and DERL are given in Table 2, in the term of average NFE, and average error of 50 runs.

Table 3, it can be seen that DEwB-1 and DEwB-2 take less NFE with respect to DE, TDE and DERL to reach the given accuracy for each benchmark problem except F_3, F_4, and F_5.

In the case of F_3 and F_5, DERL takes 212550 and 263050 NFEs respectively while DEwB-1 takes 441110 and 500000 NFE (do not reach to VTR in case of F_5) respectively and DEwB-2 takes 233160 and 299500 NFE respectively and hence DERL leads both DEwB-1 and DEwB-2. In the case of F_4, only DE successfully reaches to accuracy and hence leads all other algorithms.

Table 3, it can be observed that, in total 9 cases (F_1, F_2, F_6 - F_8, F_{10}, - F_{13}), DEwB-2 takes first place while DEwB-1 got second place. In the cases of F_3 and F_5, DEwB-2 got second position. DEwB-1 leads over all other algorithms only in the case of F_9. Both DEwB-1 and DEwB-2 are failed to achieve accuracy in two cases (F_4, F_5 for DEwB-1 and F_4, F_9 for DEwB-2).

Table 4, results are shown in term of Success Rate (SR) and Acceleration Rate (AR). From the table, it is clear that both DEwB-1 and DEwB-2 perform fast rather than DE, TDE and DERL for every benchmark problems except F_4. The average acceleration rate of DEwB-1 is 51.31% while average acceleration rate of DEwB-2 is 57.31% with respect to DE. In the same way the average AR of DEwB-1 is 30.98% and AR of DEwB-2 is 38.91% over TDE and 33.37% and 37.09% over DERL by DEwB-1 and DEwB-2 respectively.

Also the success rate of each algorithm is given in Table 4 and it can be seen that both DEwB-1 and DEwB-2 fails to achieve accuracy in two cases (F_4, F_5 for DEwB-1 and F_4, F_9 for DEwB-2) and hence success rate is 0 in these cases. If we want to see overall better average success rate then from the last column of Table 2, it is clear that DEwB-1 gives 82% success rate while DEwB-2 gives 80% success rate.

Hence from Table 3 and 4, it is observed that both DEwB-1 and DEwB-2 gives faster performance in term of NFE, AR and SR and prove their efficiency over DE, TDE and DERL.

Table 2. Benchmark functions with their global value

F	Test Function	Initial range	Min f_i
F_1	$\sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	0
F_2	$\sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$	0
F_3	$\sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^D$	0
F_4	$\max_i \{ \lvert x_i \rvert \}$	$[-100, 100]^D$	0
F_5	$\sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^D$	0
F_6	$\sum_{i=1}^{D} \lfloor x_i + 0.5 \rfloor^2$	$[-100, 100]^D$	0
F_7	$\sum_{i=1}^{D} \dot{x}_i^4 + rand(0\ 1)$	$[-1.28, 1.28]^D$	0
F_8	$\sum_{i=1}^{D} -x_i \sin\sqrt{\lvert x_i \rvert} + 418.982887D$	$[-500, 500]^D$	0
F_9	$10D + \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) \right)$	$[-5.12, 5.12]^D$	0
F_{10}	$-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=0}^{D} x_i^2} \right) - \exp\left(\sqrt{\frac{1}{D}\sum_{i=0}^{D} \cos(2\pi x_i} \right) + 20 + e$	$[-32, 32]^D$	0
F_{11}	$\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	$[-600, 600]^D$	0
F_{12}	$\frac{\pi}{D}\left\{ 10\sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_D - 1)^2 \right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ where $y_i = 1 + \frac{1}{4}(x_i + 1)$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m \text{ if } x_i > a \\ 0 \text{ if } -a \le x_i \le a \\ k(-x_i - a)^m \text{ if } x_i < -a \end{cases}$	$[-50, 50]^D$	0
F_{13}	$0.1\left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{D} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_{i+1}) \right] + (x_D - 1)^2 \left[1 + \sin^2(2\pi x_D) \right] \right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$	0

Table 3. Comparison of DEwB-1 and DEwB-2 with DE, TDE and DERL in the term of average NFE and average error of 50 runs

Fun	DE	TDE	DERL	DEwB-1	DEwB-2
F_1	104650	61700	54880	*42220*	**34510**
	(8.71E-09)	(8.74E-09)	(8.94E-09)	(8.11E-09)	**(8.75E-09)**
F_2	175120	98930	92210	*61470*	**48080**
	(9.68E-09)	(9.40E-09)	(8.75E-09)	(9.09E-09)	**(9.65E-09)**
F_3	416730	285950	**212550**	441110	*233160*
	(9.09E-09)	(9.38E-09)	**(9.00E-09)**	(9.71E-09)	(9.64E-09)
F_4	**347000**	500000	*500000*	500000	500000
	(4.98E-09)	(6.68E-01)	(8.08E-07)	(3.85E+00)	(4.23E-02)
F_5	444550	417370	**263050**	500000	*299500*
	(8.78E-09)	(9.34E-09)	**(9.02E-09)**	(1.79E+01)	(9.59E-09)
F_6	32680	19460	21470	*12410*	**10380**
	(0.0E+00)	(0.0E+00)	(0.0E+00)	(0.0E+00)	**(0.0E+00)**
F_7	200390	55780	109160	*32660*	**23260**
	(8.71E-03)	(8.87E-03)	(7.84E-03)	(6.71E-03)	**(7.97E-03)**
F_8	500000	500000	500000	*194550*	**118100**
	(2.20E+03)	(4.03E+03)	(9.27E+02)	(8.54E-09)	**(9.66E-09)**
F_9	500000	*427400*	500000	**169960**	500000
	(5.23E+01)	(9.24E-09)	(3.97E+00)	(9.94E-09)	(2.98E+00)
F_{10}	161580	92340	103360	*65060*	**51790**
	(9.50E-09)	(9.32E-09)	(9.08E-09)	(9.21E-09)	**(9.71E-09)**
F_{11}	107800	62178	70210	*43440*	**35230**
	(9.28E-09)	(9.43E-09)	(8.62E-09)	(8.67E-09)	**(7.91E-09)**
F_{12}	93610	56050	64110	*35420*	**29800**
	(8.90E-09)	(9.00E-09)	(8.91E-09)	(9.16E-09)	**(8.10E-09)**
F_{13}	102710	64140	69210	*39810*	**33190**
	(9.11E-09)	(9.05E-09)	(8.77E-09)	(7.71E-09)	**(8.81E-09)**

Table 4. Comparison of DEwB-1 and DEwB-2 with DE, TDE and DERL in the term of Success rate and Acceleration rate. Here 1,2,3,4 and 5 denotes DE, TDE, DERL, DEwB-1 and DEwB-2 respectively and 4/1, 4/2 and 4/3 means DEwB-1vs DE, DEwB-1 vs TDE and DEwB-1 vs DERL respectively. 5/1, 5/2 and 5/3 means DEwB-2vs DE, DEwB-2 vs TDE and DEwB-2 vs DERL respectively.

Fun	Success rate (%)					Acceleration rate (%)					
	1	**2**	**3**	**4**	**5**	**4/1**	**5/1**	**4/2**	**5/2**	**4/3**	**5/3**
F_1	1	1	1	1	1	59.55	**67.02**	31.57	**44.06**	23.06	**37.11**
F_2	1	1	1	1	1	64.89	**72.54**	37.86	**51.39**	33.33	**47.85**
F_3	1	1	1	1	1	--	**44.05**	--	**18.46**	--	--
F_4	0.24	0	0	0	0	--	--	0	0	0	0
F_5	0.80	1	1	0	0.90	--	**32.62**	--	**28.24**	--	--
F_6	1	1	1	1	1	56.13	**63.30**	36.22	**46.65**	42.19	**51.65**
F_7	1	1	1	1	1	83.70	**88.39**	41.44	**58.30**	70.08	**78.69**
F_8	0	0	0	1	0.64	61.09	**76.38**	61.09	**76.38**	36.70	**61.57**
F_9	0	0.20	0	0.70	0	**66.01**	0	**60.23**	--	**66.00**	0
F_{10}	1	1	1	1	1	59.73	**67.94**	29.54	**43.91**	37.05	**49.89**
F_{11}	1	1	1	1	0.94	59.70	**67.31**	30.13	**43.34**	38.12	**49.82**
F_{12}	1	1	1	1	1	62.16	**68.16**	36.80	**46.83**	44.75	**53.51**
F_{13}	1	1	1	1	1	61.24	**67.68**	37.93	**48.25**	42.47	**52.04**
Avg	0.77	0.78	0.76	**0.82**	0.80	51.58	**57.31**	30.98	**38.91**	33.37	**37.09**

6.2 Comparison with DEahcSPX, ODE and LeDE

In order to check the efficiency of DEwB-1 and DEwB-2 over some other modified variants of DE, a comparison of DEwB-1 and DEwB-2 with DEahcSPX, ODE and LeDE is given in this section,. The comparison is simulated in Table 5 and 6 in term of NFE and AR respectively. The results for DEahcSPX, ODE and LeDE are taken from[13].

From the Table 5, it observed that, DEwB-2 leads in 8 cases and got second place in 2 cases, while DEwB-1 got second place in 8 cases and leads in one cases. LeDE and ODE takes less NFE in 3 and cases respectively.

Similar performance of all variants is given in Table 6 in term of AR.

6.3 Comparison of DEwB-1 and DEwB-2 with jDE, SaDE and JADE

In Table 7 comparison is given in term of average error and standard deviation of 50 runs. Here all parameter settings and Max-NFE has been taken on the basis of[23]. From the table it can see that when compare to jDE, DEwB-1 and DEwB-2 gives less error in the total cases of 8 and 10 benchmark problems respectively while in 2 cases, both DEwB-1 and DEwB-2 gives similar performance like jDE.

Table 5. Comparison of DEwB-1 and DEwB-2 with DEahcSPX, ODE and LeDE in the term of average NFE of 50 runs

Fun	DEahcSPX	ODE	LeDE	DEwB-1	DEwB-2
F_1	80371	67524	49494	*42220*	**34510**
F_2	121695	140170	77464	*61470*	**48080**
F_3	358081	489210	**140176**	441110	*233160*
F_4	432319	**145880**	157499	NA	NA
F_5	360646	NA	**282972**	NA	299500
F_6	28095	25008	17123	*12410*	**10380**
F_7	82875	60230	33302	*32660*	**23260**
F_8	137125	147472	**111013**	194550	*118100*
F_9	209903	190604	187813	**169960**	NA
F_{10}	126294	106694	76111	*65060*	**51790**
F_{11}	85085	79888	50579	*43440*	**35230**
F_{12}	72429	63710	41384	*35420*	**29800**
F_{13}	81787	72395	50720	*39810*	**33190**

In the comparison of SaDE, DEwB-1 and DEwB-2 gives better performance in the total cases of 8 and 9 benchmark problems respectively and for 3 cases of DEwB-1 and for 2 cases of DEwB-2, both DEwB-1 and DEwB-2 perform similar to SaDE.

Table 6. Comparison of DEwB-1 and DEwB-2 with DEahcSPX, ODE and LeDE in the term of Acceleration rate. Here 4/1, 4/2 and 4/3 means DEWB-1vs DEahcSPX, DEWB-1 vs ODE and DEwB-1 vs LeDE respectively. 5/1, 5/2 an 1d 5/3 means DEwB-2vs DEahcSPX, DEwB-2 vs ODE and DEwB-2 vs LeDE respectively

Fun	4/1	5/1	4/2	5/2	4//3	5/3
F_1	47.46	57.06	37.47	48.89	14.69	30.27
F_2	49.48	60.49	56.14	65.69	20.64	37.93
F_3	--	34.88	9.83	52.33	--	--
F_4	--	--	--	--	--	--
F_5	--	16.95	--	--	--	-5.84
F_6	55.82	63.05	--	58.49	27.52	39.37
F_7	60.59	71.93	45.77	61.38	1.92	30.15
F_8	--	13.87	--	19.91	--	--
F_9	19.02	--	10.83	--	9.50	--
F_{10}	48.48	58.99	39.02	51.45	14.51	31.95
F_{11}	48.94	58.59	45.62	55.90	14.11	30.34
F_{12}	51.09	58.85	44.40	53.22	14.41	27.99
F_{13}	51.32	59.41	45.01	54.15	21.51	34.56

'--' indicate that the DEwB gives slow convergence speed

From the table, it can easily see that JADE gives better performance in the comparison of both DEwB-1 and DEwB-2. JADE gives minimum error in total cases of 8 and 7 cases with respect to DEwB-1 and DEwB-2 respectively while DEwB-1 and DEwB-2 gives better performance than JADE only in 3 and 5 cases respectively.

6.4 Statistical Analysis

In order to verify the significance of the results of DEwB-1 and DEwb-2 comparison to other considered enhanced DE variants, a non-parametric statistical analysis is performed in this section.

So as to check the global difference between the results of Table 3,5 and 7, Friedman test[37] is applied and the statistical results are simulated in Table 8,10 and 12 respectively. We can see that the *p*-value obtain by Friedman test is smaller than the significance level which are taken as $\alpha = 0.05$ and 0.1. Hence we can conclude that among the experiential results there are some significant differences.

Table 7. Comparison of DEwB-1 and DEwB-2 with jDE, SaDE and JADE in the term of average NFE of 50 runs

Fun	Max-NFE	jDE	SADE	JADE	DEwB-1	DEwB-2
F_1	150000	2.5E–28	4.5E–20	**1.8E–60**	2.7E–40	2.59E–50
		(3.5E–28)	(6.9E–20)	**(8.4E–60)**	(2.7E–40)	(5.39E–50)
F_2	200000	1.5E–23	1.9E–14	1.8E–25	7.2E–31	**2.6E–39**
		(1.0E–23)	(1.05E–14)	(8.8E–25)	(2.8E–31)	**(1.9E–39)**
F_3	500000	5.2E–14	9.0E–37	**5.7E–61**	9.1E–10	2.4E–21
		(1.1E–13)	(5.43E–36)	**(2.7E–60)**	(1.1E–09)	(4.3E–21)
F_4	500000	1.4E–15	7.4E–11	**8.2E–24**	3.79E+00	4.2E-02
		(1.0E–15)	(1.82E–10)	**(4.0E–23)**	(8.8E+00)	(4.0E–01)
F_5	300000	1.3E+01	2.1E+01	8.0E–02	2.01E+01	**3.59E–10**
		(1.4E+01)	(7.8E+00)	(5.6E–01	(1.89E+00)	**(5.4E–10)**
F_6	150000	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
		(0.0E+00)	**(0.0E+00)**	**(0.0E+00)**	**(0.0E+00)**	**(0.0E+00)**
F_7	300000	3.3E–03	4.8E–03	6.4E–04	1.1E–03	**6.2E–04**
		(8.5E–04)	(1.2E–03)	(2.5E–04)	(2.8E–04)	**(5.1E–04)**
F_8	900000	1.1E–10	1.1E–10	1.1E–10	**3.2E–12**	3.4E–12
		(0.0E+00)	(0.0E+00)	(0.0E+00)	**(5.4E–13)**	(6.5E–11)
F_9	500000	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.9E+00
		(0.0E+00)	**(0.0E+00)**	**(0.0E+00)**	**(0.0E+00)**	(1.7E+00)
F_{10}	50000	3.5E–04	2.7E–03	**8.2E–10**	1.8E–06	2.8E–08
		(1.0E–04)	(5.1E–04)	**(6.9E–10)**	(4.2E–07)	(9.2E–09)
F_{11}	50000	1.9E–05	**7.8E–04**	9.9E–08	2.4E–10	3.5E–13
		(5.8E–05)	**(1.2E–03)**	(6.0E–07)	(2.9E–10)	(4.3E–13)
F_{12}	50000	1.6E–07	**1.9E–05**	**4.6E–17**	5.1E–13	4.2E–16
		(1.5E–07)	**(9.2E–06)**	**(1.9E–16)**	(3.6E–13)	(3.8E–16)
F_{13}	50000	1.5E–06	6.1E–05	**2.0E–16**	1.2E–11	4.3E–15
		(9.8E–07)	(2.0E–05)	**(6.5E–16)**	(1.1E–11)	(3.9E–15)

Table 8. Results on Friedman test based on NFE given in Table 3

N	Friedman value	df	*p*-value
13	24.998	4	<0.001

A post-hoc statistical testing is made by using these results to distinguish concrete differences between algorithms. Firstly, Bonferroni-Dunn's test[26,38] is employed to identify significant differences for the control algorithms DEwB-1 and DEwB-2. As achieve by Critical Difference (CD) by Bonferroni-Dunn's method and Friedman's test, the ranking of algorithms are given in Table 9,11 and 13 based on their error values as presented in Table 3,5 and 7 respectively. Bonferroni-Dunn's method to calculated CD value is given in appendix I.

Table 9. Ranking achieved by Friedman test and Critical Difference (CD) calculated by Bonferroni-Dunn's method for Table 3

Algorithms	DE	TDE	DERL	DEwB-1	DEwB-2	CD at α=0.1	CD at α=0.05
Ranking	4.38	3.19	3.35	2.50	1.58	1.3898	1.5492

Table 10. Results on Friedman test based on NFE given in Table 5

N	Friedman value	Df	*p*-value
13	21.969	4	<0.001

Table 11. Ranking achieved by Friedman test and Critical difference (CD) calculated by Bonferroni-Dunn's method for Table 5

Algorithms	DEahcSPX	ODE	LeDE	DEwB-1	DEwB-2	CD at α=0.1	CD at α=0.05
Ranking	4.23	3.88	2.38	2.69	1.81	1.3898	1.5492

Table 12. Results on Friedman test based on NFE given in Table 7

N	Friedman value	Df	*p*-value
13	20.071	4	<0.001

Table 13. Ranking achieved by Friedman test and Critical difference (CD) calculated by Bonferroni-Dunn's method for Table 7

Algorithms	jDE	SaDE	JADE	DEwB-1	DEwB-2	CD at α=0.1	CD at α=0.05
Ranking	3.58	4.15	1.96	3.08	2.23	1.3898	1.5492

Figure 2(a), 2(b) and 2(c), Bonferroni-Dunn's graphic demonstrate variation between rankings attain by algorithms. In the Figure 2, a horizontal cut line is drawn that represents the threshold for the best performing algorithm with the least ranking bar. At each significance level taken in the paper (i.e. 5% and 10%) a cut line is sketched at height equal to the summation of the critical difference and ranking of the control algorithm. The bars go beyond this line are associated to an algorithm with inferior performance than the control algorithm.

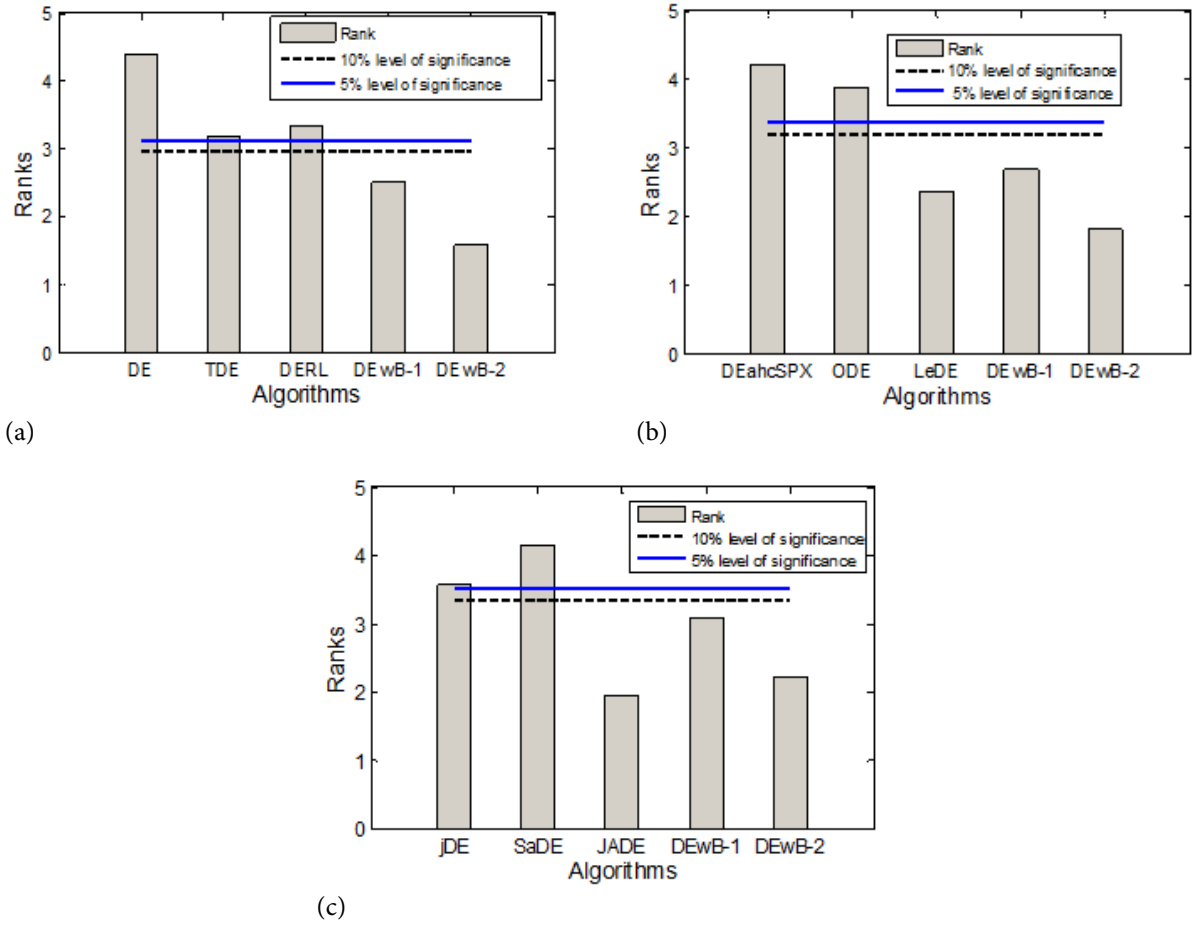


Figure 2. Bonferroni Dunn bar chart. The Bar present rank of correspondence algorithm and Horizontal cut lines shows the significant level.

In conclusion, using the application of Bonferroni-Dunn's test, it can be see that both DEwB-1and DEwB-2 gives significant performances in comparison to the other enhanced variants of DE. From the Figure 2(a), it may be notice that the DEwB-2 has performed better than the DE, TDE and DERL while in case of DEwB-1 there is no significant difference in the results. Similarly from Figure 2(b), it can be easily observed that LeDE, DEwB-1 and DEwB-2 performed equally and the performance of DeahcSPX and ODE are worse in comparison to these algorithms. From Figure 2(c), it is clear that the performances of JADE, DEwB-1 and DEwB-2 are same but the performances of jDE and SaDE are worse. All these tests are conducted at the 5% and 10% significance levels.

7. Numerical Results for Molecular Potential Energy Problem

The simulated results for molecular potential energy problem at different dimension are presented in Table 14. The results are obtained in term of average NFE, mean fitness value, Success Rate (SR) and mean CPU time of 50 runs. The best results are represented as boldface.

From the Table 14 it can be easily observe that for D=10, the average NFE taken by DE, TDE and DERL are 43970, 37980 and 28120 respectively while the average NFE taken by DEwB-1 and DEwB-2 are 23220 and 12320 respectively which are less than other algorithms.

In the same way we can see that for D=15, 20 and 25, the NFE obtained by DE are 110875, 209533 and 297840 respectively while the NFE obtained by TDE are 87380, 157010 and 228300 respectively and the NFE obtained by DERL are 62950, 82350 and 127330 respectively. The NFE obtained by DEwB-1 and DEwB-2 are 46090, 79610 and 117090 while the NFE taken by DEwB-2 are 34680, 60950 and 75600 for D=15, 20 and 25 respectively. So it is easily observed that DEwB-2 has obtained fewer NFE in the comparison of all other considered variants of DE while DEwB-1 has taken second place in each case of D=10, 15, 20 and 25 and hence both DEwB-1 and DEwB-2 proved the competence over DE, TDE and DERL.

The AR of DEwB-1 and DEwB-2 over DE, TDE and DERL are shown in Table 15 for each D=10, 15, 20 and 25. For D=10, the AR of DEwB-1 are 47.19%, 38.86% and 17.42% over DE, TDE and DERL respectively while for D=10, the AR of DEwB-2 over DE, TDE and DERL are 71.98%, 67.56% and 56.18% respectively which shows that DEWB-2 gives fast acceleration rate over DE, TDE and DERL compare to DEwB-1. Similar results may be seen from the Table 16 for D=15, 20 and 25.

Similar performance for CPU time can be analysis

Table 14. Numerical results and comparison for molecular potential energy problem of DEwB-1 and DEwB-2 with DE and TDE algorithm

Dim	Terms	DE	TDE	DERL	DEwB-1	DEwB-2
D=10	NFE	43970	37980	28120	23220	**12320**
	Fitness	-0.5893	-0.5893	-0.5893	-0.5893	-0.5893
	CPU time (Sec)	0.35	0.24	0.24	0.20	**0.12**
	SR%	100%	100%	100%	100%	100%
D=15	NFE	110875	87380	62950	46090	**34680**
	fitness	-0.4933	-0.4933	-0.4933	-0.4934	-0.4934
	CPU time (Sec)	1.50	1.20	0.80	0.30	**0.20**
	SR%	100%	100%	100%	100%	100%
D=20	NFE	209533	157010	82350	79610	**60950**
	Fitness	-1.0005	-1.0005	-1.0005	-1.0005	-1.0005
	CPU time (Sec)	2.7	1.8	0.8	0.8	**0.6**
	SR	92%	98%	100%	100%	**100%**
D=25	NFE	297840	228300	127330	117090	**75600**
	Fitness	-0.9045	-0.9045	0.9045	-0.9045	-0.9045
	CPU Time (Sec)	4.4	3.6	2.2	1.4	**0.8**
	SR	84%	94%	96%	**100%**	94%

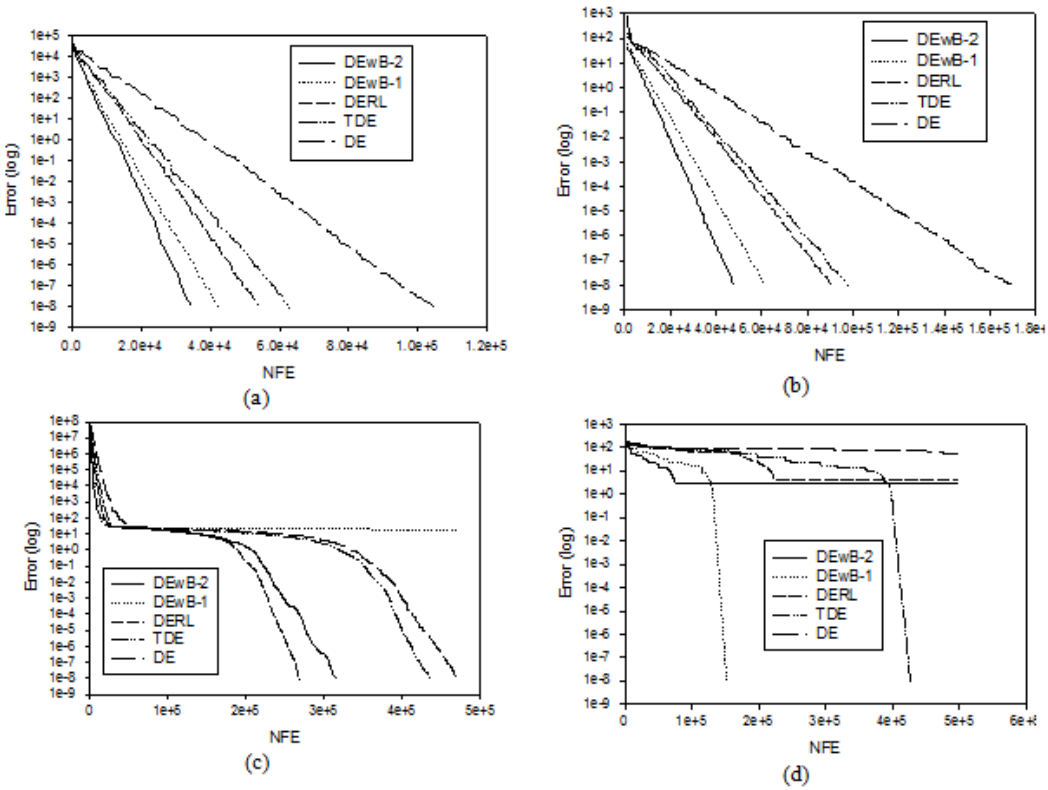
Table 15. Acceleration rate (AR) for DEwB-1 and DEwB-2 over DE, TDE and DERL for molecular energy problems

Dim	Terms	DEwB-1 *v/s*			DEwB-2 *v/s*		
		DE	TDE	DERL	DE	TDE	DERL
D=10	NFE	47.19	38.86	17.42	**71.98**	**67.56**	56.18
	CPU Time	42.85	16.66	16.67	**65.71**	**50**	**50**
D=15	NFE	58.43	47.25	26.78	**68.72**	**60.31**	**44.90**
	CPU Time	80.00	75.00	62.50	**86.66**	**83.33**	**75**
D=20	NFE	62.00	49.29	3.32	**70.91**	**61.18**	**25.98**
	CPU Time	70.37	55.56	0.00	77.77	**66.67**	**25**
D=25	NFE	60.68	48.71	8.04	**74.61**	**66.89**	**40.62**
	CPU Time	68.18	61.12	36.37	**81.81**	**77.78**	**63.64**

from Table 14 and 15. We can see that for each case of *D*=10, 15, 20 and 25, both DEwB-1 and DEwB-2 obtained the optimal solution very quickly rather than other algorithms time. The AR of DEwB-1 with respect to DE are 42.85%, 80.00%, 70.37% and 68.18%, with respect to TDE are 16.66%, 75.00%, 55.56% and 61.12% and with respect to DERL are 16.67%, 62.50%, 0.00% and 36.37% for *D*=10, 15, 20 and 25 respectively, while AR of DEwB-2 with respect to DE are 65.71%, 86.66%, 77.77% and 81.81%, with respect to TDE are 50%, 83.33%, 66.77% and 78.78% and with respect to DERL are 50%, 25%,75% and 63.64% for *D*=10,15,20 and 25 respectively. Hence DEwB-2 is again shown its efficiency over all other variants in term of CPU time also.

7.1 Convergence Graphs

Figure 3 and 4 represented the convergence graphs for benchmark problems and potential molecular energy problem respectively. The convergence graphs are obtained in term of fitness values with respect to their correspondence NFE values. From the convergence graphs we can see that for all problems, initially all algorithms having almost equal fitness values but as the NFEs increases, both DEwB-1 and DEwB-2 gained fast convergence rate over all other algorithms and hence proved the efficiency in term of convergence speed.



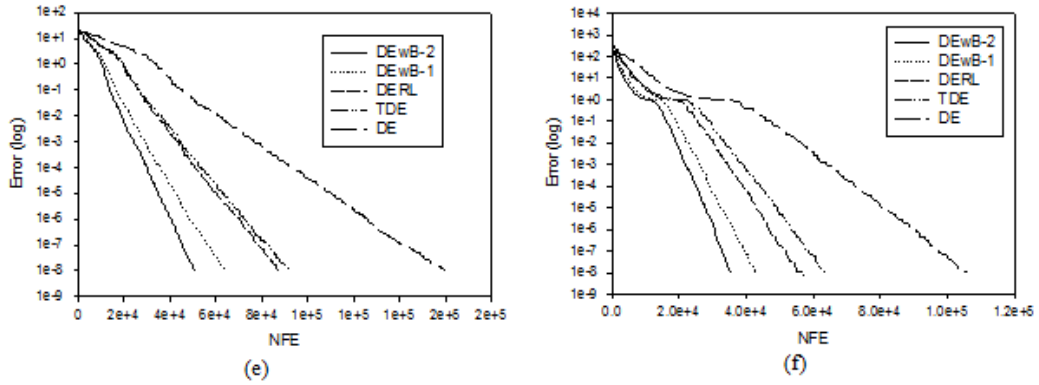


Figure 3. BConvergence graphs of function: (a) F_1 (b) F_2 (c) F_5 (d) F_9 (e) F_{10} (f) F_{11}

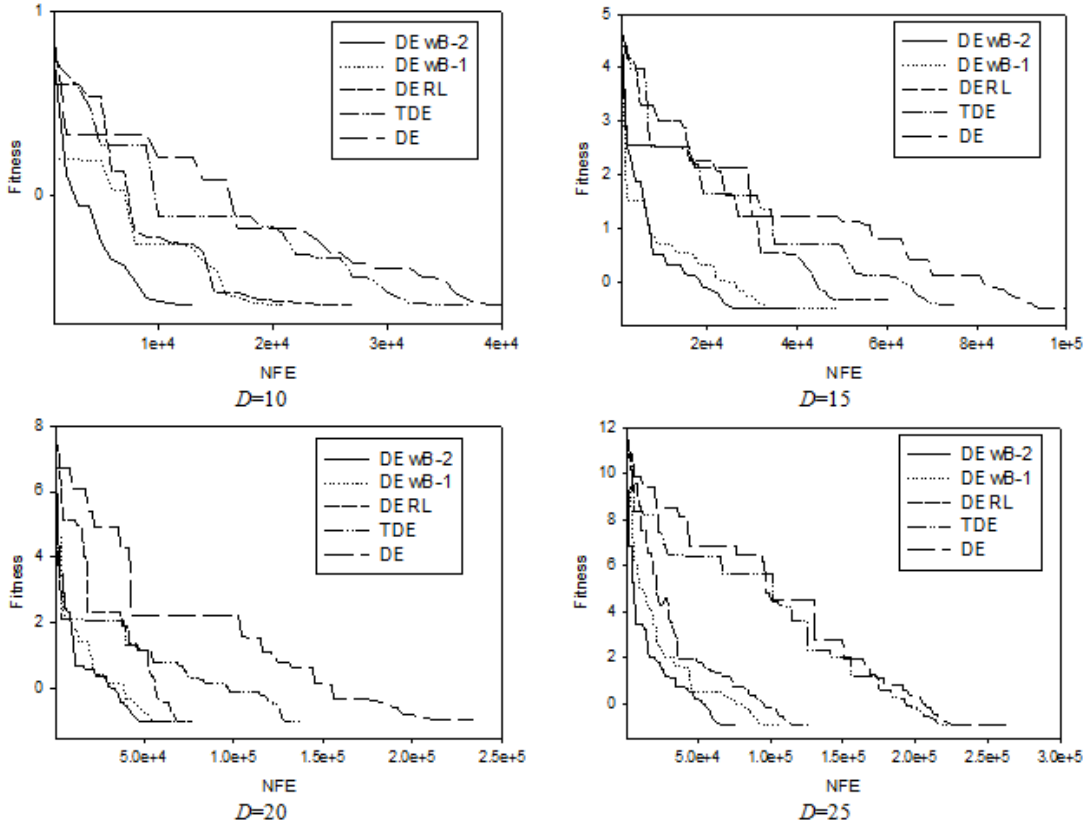


Figure 4. Convergence graph for molecular potential energy problem at D=10, 15, 20, 25.

8. Conclusion

Two modifications in DE algorithm is DEwB-1 and DEwB-2is presented in the paper. These modified algorithms are analyzed on a set of 13 standard, unconstrained benchmark functions for high dimensions. Various performance measures used for analysing the algorithms indicate that the proposed modifications facilitate in improving the performance of original DE algorithm. The proposed algorithms are applied on a real life application to determine the molecular potential energy modelled as a nonlinear optimization problem. From the numerical and statistical results, it was observed that the proposed variants DEwB-1 and DEwB-2 performed better than the original DE algorithm and other improved variants of DE algorithm. Encouraged by its performance over benchmark and molecular potential energy problems, the future plan is to apply these variants on more complicated

real life problems. Also, the study of its mathematical properties and theoretical analysis may be considered as a future research direction.

9. Acknowledgment

The author acknowledges the financial support by University Grant Commission (F.No. 30-317/2016), New Delhi, India to carry out the reported research work.

10. References

1. Storn, R., Price, K. Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization.1997 Dec; 11(4):341–59. Crossref
2. Plagianakos, V, Tasoulis D, Vrahatis M. A Review of Major Application Areas of Differential Evolution. In: Advances in Differential Evolution, Springer, 2008, 143, p 197–238. Crossref
3. Kumar S, Kumar P, Sharma TK, Pant M. Bi-Level Thresholding using PSO, Artificial Bee Colony and MRLDE Embedded with Otsu Method, Memetic Computing. 2013; 5(4):323–34. Crossref
4. Kumar P, Pant M. Noisy Source Recognition in Multi Noise Plants by Differential Evolution. In: Proceeding of SIS. 2013, p. 271–75. Crossref
5. Kumar P, Pant M, Singh VP. Modified Random Localization based de for Static Economic Power Dispatch with Generator Constraints, International Journal of Bio-Inspired Computation. 2014; 6(4):250–61. Crossref
6. Kumar S, Pant M, Ray AK. DE-IE: Differential Evolution for Color Image Enhancement, International Journal of System Assurance Engineering and Management. 2014; 1–12.
7. Ali M, Ahn CW. An Optimized Watermarking Technique based on Self-Adaptive DE in DWT-SVD Transform Domain, Signal Processing. 2014; 94:545–56. Crossref
8. AliM., Ahn CW. Pant, M. Multi-Level Image Thresholding by Synergetic Differential Evolution, Applied Soft Computing. 2014; 17:1–11. Crossref
9. Wang H, Wu Z, Rahnamayan S. Enhance Opposition based Differential Evolution for Solving High Dimensional Continuous Optimization Problems, Soft Computing. 2010; 15(11):2127–40. Crossref
10. Fan HY, Lampinen J. A Trigonometric Mutation Operation to Differential Evolution, Journal of Global Optimization. 2003; 27:105–12. Crossref
11. Noman N, Iba H. Accelerating Differential Evolution using an Adaptive Local Search, IEEE Transaction of Evolutionary Computing. 2008; 12(1):107–25. Crossref
12. Ali M, Pant M, Abraham A. Simplex Differential Evolution, Acta Polytechnica Hungarica. 2009; 6(5):95–115.
13. Cai Y, Wang J, Yin J. Learning-Enhanced Differential Evolution for Numerical Optimization, Springer-Verlag, Soft Computing. 2011; 16(2):303–30. Crossref
14. Wang Y, Cai Z, Zhang Q. Enhancing the Search Ability of Differential Evolution through Orthogonal Crossover, Information Sciences. 2012; 185:153–77. Crossref
15. Zaheer H. A New Guiding Force Strategy for Differential Evolution, International Journal of System Assurance Engineering and Management. 2015; 1–15.
16. Liu J, Lampinen J. A Fuzzy Adaptive Differential Evolution Algorithm, Soft Comput. Fusion Found Methodology Application. 2005; 9(6):448–62. Crossref
17. Ali, MM. Differential Evolution with Preferential Crossover, European Journal of Operational Research. 2007; 181:1137–47. Crossref
18. Rahnamayan S, Tizhoosh Salama M. Opposition based Differential Evolution, IEEE Transaction on Evolutionary Computations. 2008; 12(1):64–79. Crossref
19. Das S, Abraham A, Chakraborty U, Konar A. Differential Evolution using a Neighborhood based Mutation Operator, IEEE Transaction on Evolutionary Computations. 2009; 13(3):526–53.
20. Qin AK, Huang VL, Suganthan PN. Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization, IEEE Transactions on Evolutionary Computations. 2009; 13(2):398–417. Crossref
21. Zhang J, Sanderson A. JADE: Adaptive Differential Evolution with Optional External Archive', IEEE Transactions on Evolutionary Computations. 2009; 13(5):945–58. Crossref
22. Jia D, Zheng G, Khan MK. An Effective Memetic Differential Evolution Algorithm based on Chaotic Local Search, Information Sciences. 2011; 181:3175–87. Crossref
23. Mallipeddi R, Suganthan PN, Pan QK, Tasgetiren MF. Differential Evolution Algorithm with Ensemble of Parameters And Mutation Strategies, Applied Soft Computing. 2011; 11:1679–96. Crossref
24. Kumar P, Pant M. Enhanced Mutation Strategy for Differential Evolution. Proceeding of IEEE Congress on Evolutionary Computation (CEC-12); 2012. 1–6. Crossref
25. Sarker RA, Elsayed SM, Ray T. Differential Evolution with Dynamic Parameters Selection for Optimization Problems, IEEE Transactions on Evolutionary Computation. 2014; 18(5):689–707. Crossref
26. Pooja Chaturvedi P, Kumar P. Control Parameters and Mutation based Variants of Differential Evolution Algorithm, Journal of Computational Methods in Sciences and Engineering. 2015; 15(4):783–800.
27. Kumar P, Pant M. Modified Single Array Selection Operation for DE Algorithm. In: Proceedings of Fifth International Conference on Soft Computing for Problem Solving, Series Advances in Intelligent Systems and Computing; 2016, 143. p 795–803. Crossref
28. Neri F, Tirronen V. Recent Advances in Differential Evolution: A Survey And Experimental Analysis, Artificial Intelligence Reviews. 2010; 33(1–2):61–106. Crossref

29. Das S, Suganthan PN. Differential Evolution: A Survey of the State-of-the-Art, IEEE Transactions on Evolutionary Computations. 2011; 15(1):4–13 Crossref
30. Brest J, Greiner S, Boskovic B, Mernik M, Zumer V. Self Adapting Control Parameters in Differential Evolution; A Comparative Study on Numerical Benchmark Problems, IEEE Transactions on Evolutionary Computation. 2006; 10(6):646–57. Crossref
31. Kaelo P, Ali MM. A Numerical Study of some Modified Differential Evolution Algorithms, European Journal of Operational Research. 2006; 169:1176–84. Crossref
32. Thangaraj R, Pant M, Abraham A. A Simple Adaptive Differential Evolution Algorithm. IEEE World Congress on Nature and Biologically Inspired Computing and 8th International Conference on Computer Information Systems and Industrial Management Applications; 2009. p. 457–62. Crossref
33. Yang Z, Tang K, Yao. Self-Adaptive Differential Evolution with Neighborhood Search. Proceeding of IEEE Congress on Evolutionary Computation, Hong Kong; 2008. p. 1110–16. PMCid:PMC2442166.
34. Fan HY, Lampinen, J, Dulikravich GS. Improvements to Mutation Donor Formulation of Differential Evolution. International Congress on Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems Eurogen; 2003.
35. Lavor C, Maculan N. A Function to Test Methods Applied to Global Minimization of Potential Energy of Molecules, Numerical Algorithms (Springer). 2004; 35:287–300. Crossref
36. Maranas CD, Floudas CA. A Deterministic Global Optimization Approach for Molecular Structure Determination, Journal of Chemical Physics (AIP). 1994; 100(2):1247–61. Crossref
37. Kumar P, Pant M, Abraham A. Two Enhanced Differential Evolution Variants for Solving Global Optimization Problems. In: Proceeding of Third IEEE World Congress on Nature and Biologically Inspired Computing (NABIC-11); 2011. 208–13. Crossref
38. Demsar J. Statiscally Comparisons of Classifier over Multiple Date Set, Journal of Machine Learning Research. 2006; 7:1–30.
39. Dunn OJ. Multiple Comparisons among Means, Journal of the American Statistical Association. 1961; 56:52–64. Crossref

Appendix I

Critical difference is use to check the significant difference between performance of two classifiers on the basis of their corresponding average ranks. As suggested by Demsar (2006), the *CD* value is calculated as:

$$CD = q\alpha\sqrt{\frac{k(k+1)}{6N}}$$

where k denotes the number of algorithms and N denotes number of benchmark problems. The value of q_α on significant level at $\alpha = 0.05$ and 0.1 is given in Table 16.

Table 16. Critical values for the two-tailed Bonferroni-Dunn test; the number of classifiers include the control classifier

Classifier	2	3	4	5	6	7	8	9	10
$q_{0.05}$	1.960	2.241	2.394	2.498	2.576	2.638	2.690	2.724	2.773
$q_{0.1}$	1.645	1.960	2.128	2.241	2.326	2.394	2.450	2.498	2.539