International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

ReDE- A Revised mutation strategy for Differential Evolution Algorithm

Meera Ramadas [1*] Ajith Abraham [2] Sushil Kumar [3]

[1] AIIT, Amity University, Uttar Pradesh, Noida, India
[2] MIR Labs, Washington, USA
[3] Department of Computer science, ASET, Amity University, Uttar Pradesh, Noida, India
*Corresponding author's Email: meera_mgr@rediffmail.com

Abstract: In today's scientific world, evolutionary algorithm is considered as a major area for research providing immense opportunity to researchers. Several works are done in this area to create different variants of the evolutionary algorithm and to further improve their performance. Differential Evolution (DE) is considered to be a dominant technique for optimization and is being used to solve various real time problems. In this paper, a revised mutation strategy has been implemented. This strategy uses two control parameters and two types of population. The results computed were compared with the existing mutation strategy with the results showing better performance for the new revised strategy.

Keywords: mutation, control parameters, recombination, generation, optimisation.

1. Introduction

Many of real time problems in the scientific world can be considered as constrained optimisation problems. For solving these optimisation problems, we can use various evolutionary algorithms. Evolutionary algorithms are those algorithms that uses the technique prompted by biological evolution such as reproduction, selection, mutation and recombination. Various evolutionary algorithms have been developed in due course of time.

In 1995, Storn and Price proposed the Differential Evolution (DE) which is a powerful and efficient algorithm to solve optimisation problems. Like various other evolutionary algorithms, DE is also a population based stochastic method. DE is among the best of evolutionary algorithm for solving real valued test functions. DE has various significant features like its ease to implement, better performance, low space complexity and fewer control parameters. The efficiency and performance of DE greatly depends on the trial vector generation strategy and the control parameters used.

Various researches are being done on improving the performance of DE by changing the control parameters and the trial vector generation strategy. In this paper we are introducing a new mutation strategy for DE named as Revised Differential Evolution (ReDE). This strategy uses two types of control parameters: a variable parameter value and a constant parameter value between range (0,1) and two types of population: the population from previous generation and new population from current generation. This mutation strategy was implemented for Differential Evolution algorithm. The results obtained for best value, number of function evaluation and CPU time were computed for different dimensions. These results were then compared to the results obtained using traditional mutation strategies. Friedman's testing was done on the values obtained and ranks for the various mutation strategies were computed. Graphs were also plotted for the values obtained. After close and comparative study, it was seen that ReDE strategy has remarkably better results in comparison to the classical DE approaches.

2. Background literature study:

DOI: 10.22266/ijies2016.1231.06

Madavan et al. [1] extended the Differential evolution approach to a multi objective optimisation problem using a Pareto based approach. The result is a simple strategy that is self - adaptive and maintains the diversity of pareto set. Babu et al. [2] used the Differential evolution approach to solve a multi objective optimisation problem with two objective functions and a classical HimmelBlau function. In both the approaches, the results obtained were better. Rene et al [3] improvised the DE algorithm with the crowding scheme. This made it possible to track and maintain the local optima. Qin et al [4] proposed a novel self – adaptive Differential evolution (SaDE) in which the control parameters need not be specified. The performance was evaluated and compared. Efren et al [5] did a comparative study on variants of Differential evolution on global optimisation. After the study, the best strategy among different variants were obtained. Hong–Kyu Kim et al. [6] gave some modifications to the differential evolution strategy for the constraint global optimisation problem. This method was applied to the design of gas circuit breaker to increase small current interruption performance. Rahnamayan et al. [7] presented a novel algorithm to accelerate differential evolution. This method employs opposition based learning for population initialization and generation jump. Qin et al. [8] proposed a self-adaptive DE algorithm where the control parameters and trial vector generation strategies are gradually self-adapted by learning from previous experience. This method eliminated time consuming search for trial vector generation and control parameters.

Takahama et al. [9] proposed the Ɛ constrained differential evolution (ƐDE) combining Differential evolution with Ɛ constrained technique. This algorithm has faster execution and high quality results. Wang et al. [10] did a study on improving the performance of DE by combining the trial vectors strategies with control parameter settings. He also proposed a method composite DE (CoDE) using three trial vectors and three control parameters. Qu et al.[11] developed neighbourhood based differential evolution in which the mutation is performed within each Euclidean neighbourhood. The mutation neighbourhood maintains the multiple optima found during evolution and progress towards the respective global/local optimum. Saber et al. [12] gave an improved Differential evolution that uses a combination of differential mutation operations. This algorithm permits a covariance adaptation matrix evolution strategy algorithm for local search. Wei-Jie Yu et al.[13] proposed an adaptive DE(ADE) with two level adaptive parameter control scheme and a new mutation strategy. This technique has the advantage of balancing between fast convergence and population diversity.

Wenyin et al. [14] devised the adaptive ranking mutation operator (ARMOR) for differential evolution. This technique is expected to make DE achieve feasible solutions faster and converge faster. This technique is simple and can be easily combined with most of constrained DE variants. Qiao et al. [15] gave an algorithm by combining the heuristics with modified Differential evolution algorithm for situations where the communication range value is unknown. His proposed method gave better results compared to other nonconvex optimization techniques.

Variants of DE are developed to improve the optimisation performance. Das et al.[16] stated that DE is defensive to the choice of control parameters making DE inconvenient to adjust to different problems.Also, DE encounters premature convergence in which searching of solution is getting trapped in local minima. Many researches are done to make DE more robust in resistant to interdependency of search variables according to Das et al.[17] .De also faces issues on non linearly separable functions. For populations clusterd in limited regions, DE has less capability to move its population across large distance in search space. Owing to these reasons, there is a need to create variants for DE that performs better in these aspects.

3. Classical Differential Evolution:

Storn and Price [18] developed the Differential Evolution (DE) algorithm whose processes are almost similar to genetic algorithm. It is a simple real parameter optimization algorithm. DE uses a population of N candidate solutions given as $X_{i,G}$ where i=1,2.. N in which index i depicts population and G shows the generation of the population. Differential Evolution algorithm depends on the three operations mainly mutation, selection and crossover.

Mutation: This operator makes DE different from other Evolutionary algorithms. It computes the weighted difference between the vectors in population. Mutation process starts by selecting three individuals at random from the population. For a given parameter $X_{i,G}$, three vectors $X_{r1,G}$, $X_{r2,G}$ and $X_{r3,G}$ are selected randomly such that r1, r2, r3 are distinct. Then the donor vector $V_{i,G}$ is computed as:

$$v_{i,G} = x_{r1,G} + F \times (x_{r2,G} - x_{r3,G}) \qquad (1)$$

Here F is the mutation factor which is a constant from [0,1].

Crossover: This process also called as recombination takes the successful solutions into the population. The trial vector $U_{i,G}$ is created for the target vector $X_{i,G}$ through binomial crossover. Elements of donor vector enter trial vector with probability C_r ε [0,1] is the crossover probability which is selected along with population size NP ≥ 4.

$$U_{j,i,G+1} = \begin{cases} V_{j,i,G+1} & if\ rand_{i,j}[0,1] \le C_r\ or\ if\ j = I_{rand} \\ X_{j,i,G+1} & if\ rand_{i,j}[0,1] > C_r\ or\ if\ j \ne I_{rand} \end{cases} \tag{2}$$

Here $rand_{i,j} \approx U[0,1]$ and I_{rand} is random integer from 1,2…N.

Selection: This operation differs from the selection operation of other evolutionary algorithms. Here the population for next generation is selected from vectors in current population and its corresponding trial vectors. The target vector $X_{i,G}$ is compared with the trial vector $V_{i,G}$ and the lowest function value is taken into next generation.

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & if\ f(U_{i,G+1}) \le f(X_{i,G})\ where\ i = 1,2,...N \\ X_{i,G} & otherwise \end{cases} \tag{3}$$

Mutation, crossover and selection operations are continued until some stopping criteria is reached.

4. Revised mutation strategy

A new strategy has been proposed for mutation called Revised DE strategy (ReDE). As it involves the best solution vector, it coincides faster as compared to the traditional strategies having random vectors only. This strategy of mutation uses two sets of population: the old population represented as popold and the new population from which the variable X_{r1}^{G} is chosen at random. Here, two parameters are being used. The parameter F known as amplifying parameter takes a constant value while the new parameter N takes a varying value which lies between (0,1).The proposed strategy is given as:

$$X' = X_{best}^{G} + N \times (X_{best}^{G} - popold) - F \times (X_{best}^{G} - X_{r1}^{G}) \tag{4}$$

5. Experimental setting

The revised algorithm was implemented on i7 core processor, 64 bit operating system with 12 GB RAM using MATLABr2008b and a comparative result was obtained with five different mutation strategy of DE algorithm. We have taken fifteen different functions and computed the results. The value to reach (VTR) is the global minimum or maximum of the function or it is a value to stop the optimisation if it is reached. We have also tabulated various results by fixing the dimension as 25 and 50. The results are tabulated for comparison with the existing algorithms.

Computed on Best Value obtained after 25 runs (vtr=1.e-015):

Table 1. Best Value obtained after 25 Runs for different functions

Function	**DE**					
	DE/best/1	**DE/rand/1**	**DE/best/1**	**De/best/2**	**DE/best/1**	**ReDE**
Sphere	9.73×10^{-16}	6.90×10^{-16}	7.53×10^{-16}	9.66×10^{-16}	7.17	**6.04×10^{-16}**
Beale	3.27×10^{-16}	**2.32** $\times 10^{-16}$	3.71×10^{-16}	7.59×10^{-16}	7.73×10^{-16}	5.95×10^{-16}
Booth	3.50×10^{-16}	2.05×10^{-16}	6.07×10^{-16}	7.08×10^{-16}	8.35×10^{-16}	**1.36×10^{-16}**
Schwefel	**-1.80×10^{3}**	**-2.25×10^{3}**	**-7.84×10^{1}**	**-1.38×10^{3}**	**-1.66×10^{3}**	**$-2.10 \times x10^{3}$**
Michlewicz	**-7.64**	**-7.21**	**-7.39**	**-6.95**	**-6.84**	**-6.6**
Schaffer N.2	6.60×10^{-16}	8.88×10^{-16}	4.43×10^{-16}	6.55×10^{-16}	8.87×10^{-16}	**$2.22\ x \times 10^{-16}$**
Schaffer N.4	3.05×10^{-15}	2.90×10^{-1}	2.92×10^{-1}	2.93×10^{-1}	2.89×10^{-1}	**2.82×10^{-1}**
HimmelBlau	1.60×10^{-16}	8.05×10^{-16}	3.83×10^{-16}	9.12×10^{-16}	**1.46×10^{-16}**	3.35×10^{-16}
Bird	**-1.04×10^{-02}**	**-1.07×10^{-2}**	**-1.05×10^{-2}**	**-1.07×10^{-2}**	**-1.03×10^{-2}**	**-1.03×10^{-2}**
Extended Cube	3.31×10^{-15}	**4.98×10^{-5}**	6.10×10^{-8}	1.93×10^{-5}	2.68	8.60×10^{-15}
Ackeley	7.19×10^{-15}	**6.46×10^{-12}**	7.99×10^{-15}	3.63×10^{-13}	3.09	1.50×10^{-14}
Gold	3.00	3.00	3.00	3.00	3.00	3.00
Griewank	9.99×10^{-16}	9.99×10^{-16}	1.60×10^{-13}	6.56×10^{-13}	1.07	**2.40×10^{-2}**
Rastrigin	1.79×10^{1}	1.23×10^{2}	7.47×10^{1}	1.28×10^{2}	1.52×10^{2}	2.98×10^{1}
Rosenbrock	9.60×10^{-16}	**1.07×10^{-8}**	7.88×10^{-16}	3.90×10^{-9}	1.07×10^{1}	1.50×10^{1}

Computed on NFE based on fixed VTR after 25 runs (vtr=1.e-015):

Table 2. NFE based on fixed VTR for different functions

Function	**DE**					
	DE/best/1	**DE/rand/1**	**DE/best-to-rand/1**	**DE/best/2**	**DE/rand/2**	**ReDE**
Sphere	288000	3705000	313000	3260000	5000000	**155000**
Beale	**48000**	94000	67000	85000	127000	50000
Booth	500000	90000	71000	77000	118000	**48000**
Schwefel	7000	12000	13000	4000	6000	**2000**
Michlewicz	1000	1000	1000	1000	1000	1000
Schaffer N.2	68000	148000	119000	139000	224000	**10000**
Schaffer N.4	5000000	5000000	5000000	5000000	5000000	5000000
HimmelBlau	**45000**	95000	67000	77000	199000	55000
Bird	1000	1000	1000	1000	1000	1000
Extended Cube	5000000	5000000	5000000	5000000	5000000	5000000
Ackeley	5000000	5000000	5000000	5000000	5000000	5000000
Gold	5000000	5000000	5000000	5000000	5000000	5000000
Griewank	2880000	4579000	500000	500000	500000	5000000
Rastrigin	5000000	5000000	5000000	5000000	5000000	5000000
Rosenbrock	**60900**	5000000	77000	5000000	5000000	5000000

Computation on elapsed time of CPU in seconds after 25 runs (vtr=1.e-015):

Table 3. Elapsed time of CPU in seconds for different functions

Function	**DE**					
	DE/best/1	**DE/rand/1**	**DE/best-to-rand/1**	**DE/best/2**	**DE/rand/2**	**ReDE**
Sphere	37.67	139.169	16.318	122.399	225.799	**16.02**
Beale	48.163	9.5786	**8.263**	8.533	8.81	32.65
Booth	11.6	8.25	6.817	8.494	8.5762	16.78
Schwefel	17.712	5.337	**4.3044**	4.956	4.5789	13.05
Michlewicz	2.73	2.217	2.817	**2.064**	2.074	6.4
Schaffer N.2	36.311	23.22	17.3	18.86	21.19	**16.7**
Schaffer N.4	190.07	227.9	261.1	238.23	245.6	360.1
HimmelBlau	14.8	16.94	18.4	**12.3**	15.95	43.2
Bird	12.39	11.8	8.25	8.71	8.07	16.64
Extended Cube	**186.78**	372.56	345.3	352.3	335.02	379.3
Ackeley	314.6	310.42	325.56	312.2	**300.4**	362.3
Gold	**257.45**	425.8	331.62	312.52	319.5	314.5
Griewank	321.7	305.3	346.58	342.76	341.01	**294.3**
Rastrigin	**254.69**	316.8	265.4	310.1	323.52	300.6
Rosenbrock	52.01	36.23	52.79	324.06	**34.53**	300.6

Computed on Best Value obtained after 25 runs (vtr=1.e-014):

Table 4. Best Value obtained after 25 Runs for different functions

Function	**DE**					
	DE/best/1	**DE/rand/1**	**DE/best-to-rand/1**	**DE/best/2**	**DE/rand/2**	**ReDE**
Sphere	9.34 $\times 10^{-15}$	9.35 $\times 10^{-15}$	9.54 $\times 10^{-15}$	9.94 $\times 10^{-15}$	6.92	$\mathbf{8.80 \times 10^{-15}}$
Beale	4.26 $\times 10^{-15}$	7.72 $\times 10^{-15}$	1.13 $\times 10^{-15}$	1.36 $\times 10^{-17}$	7.50 $\times 10^{-15}$	$\mathbf{1.19 \times 10^{-15}}$
Booth	1.81 $\times 10^{-15}$	7.55 $\times 10^{-16}$	$\mathbf{1.95 \times 10^{-15}}$	2.75 $\times 10^{-15}$	6.47 $\times 10^{-15}$	2.32 $\times 10^{-16}$
Schwefel	$\mathbf{-2.22 \times 10^{3}}$	$\mathbf{-4.80 \times 10^{2}}$	$\mathbf{-1.67 \times 10^{3}}$	$\mathbf{-4.47 \times 10^{3}}$	$\mathbf{-1.50 \times 10^{3}}$	$\mathbf{-6.80 \times 10^{2}}$
Michlewicz	**-7.69**	**-7.64**	**-6.87**	**-7.35**	**-6.98**	**-6.6**
Schaffer N.2	1.33 $\times 10^{-15}$	1.33 $\times 10^{-15}$	6.66 $\times 10^{-16}$	5.30 $\times 10^{-15}$	$\mathbf{1.33 \times 10^{-15}}$	4.40 $\times 10^{-16}$
Schaffer N.4	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$
HimmelBlau	4.83 $\times 10^{-15}$	4.42 $\times 10^{-15}$	$\mathbf{1.90 \times 10^{-15}}$	3.95 $\times 10^{-15}$	5.14 $\times 10^{-15}$	4.14 $\times 10^{-15}$
Bird	-9.30 $\times 10^{1}$	-1.04 $\times 10^{2}$	-1.07 $\times 10^{2}$	-1.03 $\times 10^{2}$	-1.04 $\times 10^{2}$	-1.03 $\times 10^{2}$
Extended Cube	5.70 $\times 10^{-6}$	5.21 $\times 10^{-5}$	7.10 $\times 10^{-8}$	$\mathbf{1.73 \times 10^{-5}}$	2.92 $\times 10^{-9}$	5.47 $\times 10^{-8}$
Ackeley	7.99 $\times 10^{-15}$	5.02 $\times 10^{-15}$	7.99 $\times 10^{-15}$	3.59 $\times 10^{-13}$	3.21	1.02
Gold	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
Griewank	$\mathbf{1.48 \times 10^{-2}}$	9.21 $\times 10^{-15}$	7.88 $\times 10^{-15}$	5.07 $\times 10^{-9}$	1.06	6.66 $\times 10^{-16}$
Rastrigin	$\mathbf{3.61 \times 10^{1}}$	**1.18** $\times 10^{2}$	$\mathbf{8.17 \times 10^{1}}$	$\mathbf{1.73 \times 10^{2}}$	$\mathbf{1.67 \times 10^{2}}$	**2.88**
Rosenbrock	3.98	1.40 $\times 10^{-8}$	6.90 $\times 10^{-15}$	1.56 $\times 10^{-11}$	7.15 $\times 10^{4}$	1.50

Computed on Best Value obtained after 50 runs (vtr=1.e-015):

Table 5. Best Value obtained after 50 Runs for different functions

Function	**DE**					
	DE/best/1	**DE/rand/1**	**DE/best-to-rand/1**	**De/best/2**	**DE/rand/2**	**ReDE**
Sphere	8.50 $\times 10^{-16}$	9.99 $\times 10^{-16}$	6.20 $\times 10^{-16}$	9.70 $\times 10^{-16}$	**2.52**$\times 10^{-1}$	9.68 $\times 10^{-16}$
Beale	5.90 $\times 10^{-16}$	2.56 $\times 10^{-16}$	3.80 $\times 10^{-16}$	9.73 $\times 10^{-16}$	3.90 $\times 10^{-16}$	**7.40**$\times 10^{-15}$
Booth	3.17 $\times 10^{-16}$	5.87 $\times 10^{-16}$	$\mathbf{1.76 \times 10^{-16}}$	5.90 $\times 10^{-16}$	3.7 $\times 10^{-16}$	4.31 $\times 10^{-16}$
Schwefel	-5.60 $\times 10^{2}$	-2.48 $\times 10^{3}$	-6.40 $\times 10^{2}$	-4.93 $\times 10^{3}$	-2.40 $\times 10^{3}$	-3.50 $\times 10^{3}$
Michlewicz	-1.13 $\times 10^{1}$	-1.20 $\times 10^{1}$	-1.21 $\times 10^{1}$	-1.24 $\times 10^{1}$	-1.12 $\times 10^{1}$	-1.17 $\times 10^{1}$
Schaffer N.2	6.66 $\times 10^{-16}$	8.88 $\times 10^{-16}$	0.00	8.88 $\times 10^{-16}$	6.66 $\times 10^{-16}$	$\mathbf{6.56 \times 10^{-16}}$
Schaffer N.4	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$	2.92 $\times 10^{-1}$
HimmelBlau	6.19 $\times 10^{-16}$	7.18 $\times 10^{-17}$	5.85 $\times 10^{-17}$	7.17 $\times 10^{-16}$	7.67 $\times 10^{-16}$	$\mathbf{3.67 \times 10^{-16}}$
Bird	-1.04 $\times 10^{2}$	-1.06 $\times 10^{2}$	-1.06 $\times 10^{2}$	-1.06 $\times 10^{2}$	-1.01 $\times 10^{2}$	-1.05 $\times 10^{2}$
Extended Cube	1.56 $\times 10^{-7}$	2.94 $\times 10^{-10}$	1.36 $\times 10^{-7}$	9.42 $\times 10^{-11}$	2.61 $\times 10^{-14}$	$\mathbf{1.02 \times 10^{-7}}$
Ackeley	7.99 $\times 10^{-15}$	6.33 $\times 10^{-13}$	7.99 $\times 10^{-15}$	9.32 $\times 10^{-14}$	$\mathbf{7.19 \times 10^{-14}}$	7.99 $\times 10^{-15}$
Gold	3.00	3.00	3.00	3.00	3.00	3.00
Griewank	**1.23** $\times 10^{-2}$	8.88 $\times 10^{-16}$	7.65 $\times 10^{-16}$	8.88 $\times 10^{-16}$	6.77 $\times 10^{-11}$	7.7 $\times 10^{-16}$
Rastrigin	6.28 $\times 10^{1}$	9.72 $\times 10^{1}$	6.57 $\times 10^{1}$	1.05 $\times 10^{2}$	1.29 $\times 10^{1}$	1.98 $\times 10^{1}$
Rosenbrock	9.50 $\times 10^{-16}$	2.23 $\times 10^{-12}$	3.90	7.50 $\times 10^{5}$	3.51 $\times 10^{8}$	1.40 $\times 10^{1}$

A comparative analysis was performed and study done on each of the technique. By setting the dimension as 25 and value-to-reach (VTR) as e-015, the best value, number of function evaluation (NFE) and the CPU time of different function strategies were calculated. It was noted that the proposed hybrid algorithm gave the best value for most of the standard functions.

By changing the VTR value to e-014 for dimension of 25, the best value was obtained for proposed algorithm for most of the standard functions. In some functions, the results were good for both classical DE and proposed algorithm. NFE for most of the standard functions were also best for the proposed algorithm. By changing the dimension to 50 and setting the VTR to e-015, best value and NFE was best for most of the standard functions using proposed algorithm. CPU time was also best for almost all standard functions using the proposed algorithm. The results shown in bold depict the best results obtained for different functions. From the results tabulated, it is clearly shown that the strategy ReDE gives good results for most of the functions used.

6. Graphical results

The above tabulated values were represented in a graphical form. The graphs show performance curve of five different function strategies. The x-axis represents the number of function evaluation and y-axis represents the objective function. The graph is plotted for the various values at each iteration for fixed VTR value of e-015 and dimension size of 25.

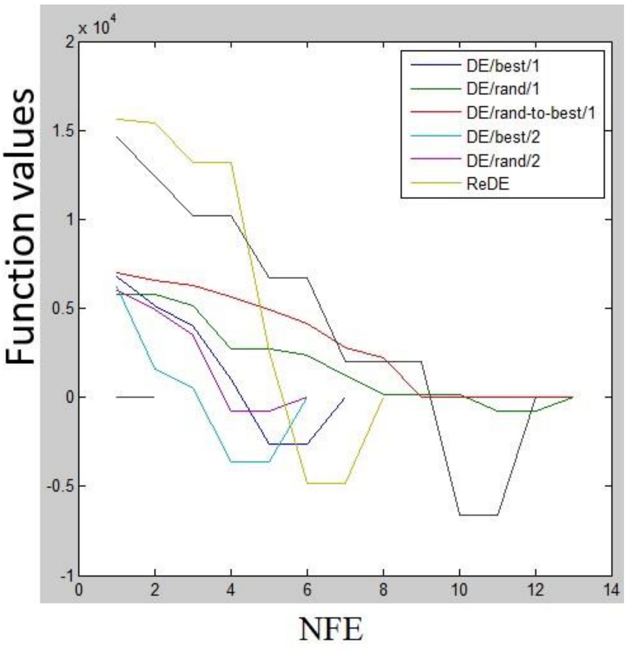

Figure.1 Graphical representation for Schwefel function

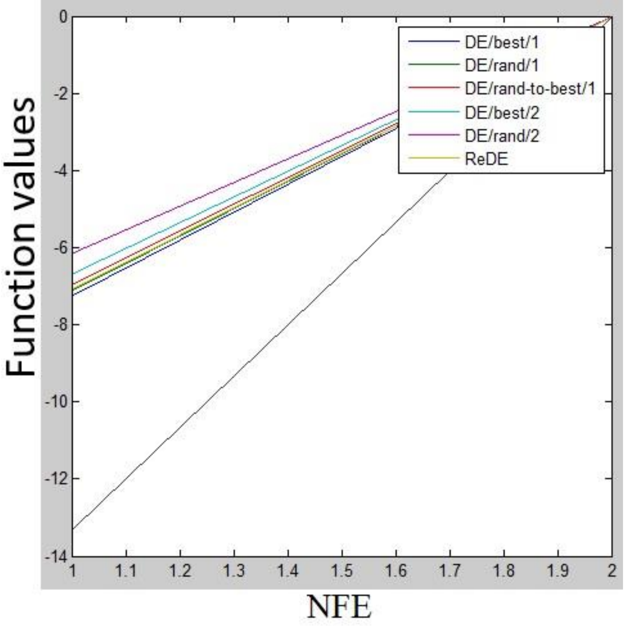

Figure.2 Graphical representation for Michelawicz function

A comparative study was done based on above graphs. The study showed that the revised mutation strategy gave better results compared to the existing mutation strategy for various functions. The graphs for schwefel and Michelawicz function show that ReDE gives different results from the traditional mutation strategy which has resulted in producing a better curve compared to the earlier techniques. The prevailing variants of DE use either two types of population or differs in the number of control parameters. As this strategy ReDE uses a combination of two sets of population with two different control parameters, the results were refined and the efficiency of the strategy was increased.

7. Statistical analysis

Based on the values from table 1, Friedman test was applied and the results obtained were tabulated. Table 6 represents the values obtained from the test and table 7 depicts the rank of the various mutation strategies used based on best value, NFE and CPU time.

Table 6. Test statistics using Friedman's test

N	25
Chi sq	22.68
Df	5
Asymptotic Significance	0.004

Table 7. Ranks of the different strategies

Strategies	**Mean Rank on Best Value**	**Mean rank on CPU time**	**Mean rank on NFE**
DE/best/1	2.7	3.4	3.0
De/rand/1	3.2	3.6	4.2
DE/best-to-rand/1	2.7	3.2	3.2
De/best/2	4.3	3.1	3.4
DE/rand/2	5.1	3.1	4.2
ReDE	3	4.4	2.86

The above tables show that the new mutation strategy has significant performance in comparison to the existing mutation strategies. The rank obtained on the basis of NFE is the best for ReDE. The rank obtained on the basis of best value is comparatively better for best value case. These rankings obtained on the basis of Freidman’s test justifies the efficiency of

ReDE strategy. Based on the ranks obtained, a graphical representation of the results is shown below. The x axis of the graph represents the various strategies used and the y axis shows the ranks obtained.

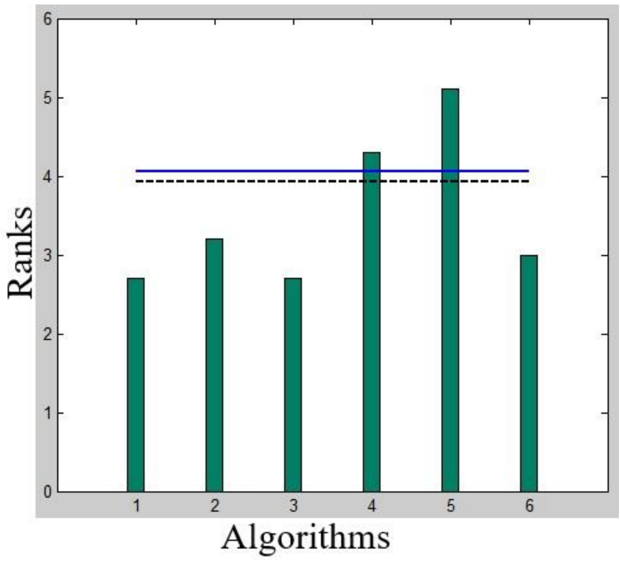

Figure 3. Bonferoni Dunn bar chart for rank on best value

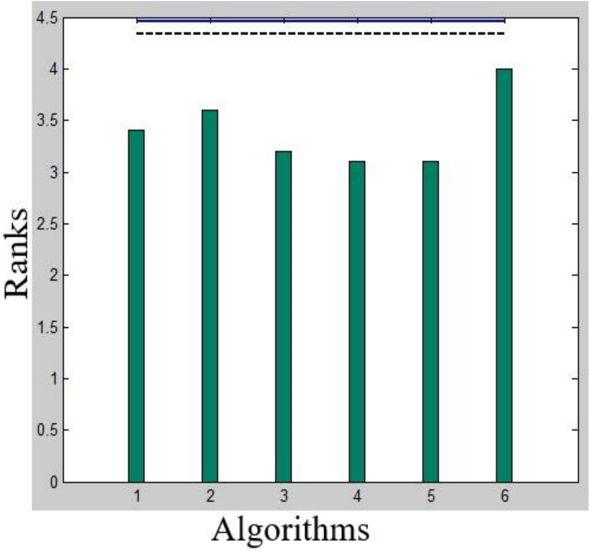

Figure 4. Bonferoni Dunn bar chart for rank based on CPU time

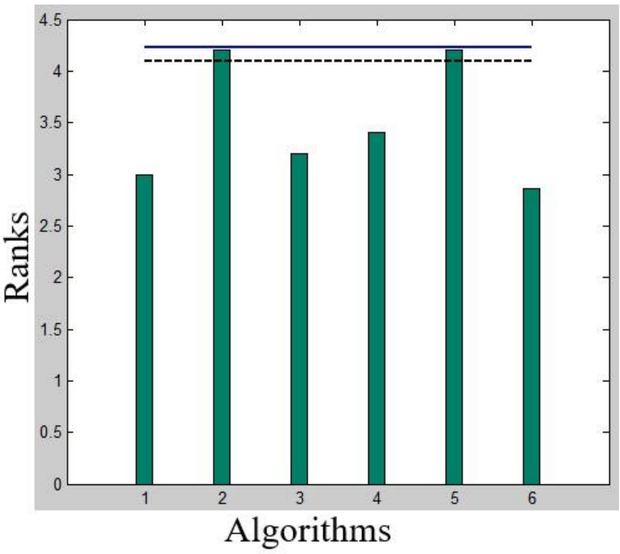

Figure 5.Bonferoni Dunn bar chart for rank based on NFE

8. Conclusion

In the above proposed work, the revised strategy was compared against the existing mutation strategies. After performing the comparative study, it was found that the proposed strategy gave much better or almost similar results for most of the functions evaluated. The ranks obtained also justifies the efficiency of the strategy. Results wre compared on the basis of VTR, CPU time and NFE. A detailed study was done and graphs were plotted. Considering all the results and details, the proposed strategy is performing better than few classical strategies. Moreover, the proposed strategy is easy and simple to apply and do not disturb the basic structure of DE. This proposed strategy can be extended to real time problems and optimisation problems. Further the work can applied to the field of clustering , image thresholding , image enhancement , image analysis, digital image processing etc. for verifying the performance of the new mutation strategy in that area.

References

[1] Madavan, Nateri K., and Bryan A. Biegel., "Multiobjective optimization using a Pareto differential evolution approach”, In*: Proceedings of the Evolutionary Computation on 2002. CEC'02. Proceedings of the 2002 Congress-Volume 02*, IEEE Computer Society, pp. 1145-1150,2002.

[2] Babu, B. V., and M. Mathew Leenus Jehan. "Differential evolution for multi-objective optimization.", *In Evolutionary Computation, 2003. CEC'03*, The 2003 Congress on, vol. 4, IEEE, pp. 2696-2703, 2003.

[3] Thomsen, Rene. "Multimodal optimization using crowding-based differential evolution.",

In Evolutionary Computation, 2004. CEC2004. Congress on, vol. 2, IEEE, pp. 1382-1389, 2004.

[4] Qin, A. Kai, and Ponnuthurai N. Suganthan. "Self-adaptive differential evolution algorithm for numerical optimization.", *In 2005 IEEE congress on evolutionary computation,* vol. 2, IEEE, pp. 1785-1791, 2005.

[5] Qin, A. Kai, and Ponnuthurai N. Suganthan. "Self-adaptive differential evolution algorithm for numerical optimization.", *In 2005 IEEE congress on evolutionary computation*, vol. 2, IEEE, pp. 1785-1791, 2005.

[6] Mezura-Montes, Efrñn, Jesús Velázquez-Reyes, and Carlos A. Coello Coello. "A comparative study of differential evolution variants for global optimization." In : *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, ACM, pp. 485-492, 2006.

[7] Kim, Hong-Kyu, Jin-Kyo Chong, Kyong-Yop Park, and David A. Lowther., "Differential evolution strategy for constrained global optimization and application to practical engineering problems.", *IEEE Transactions on Magnetics 43,* no. 4 (2007), pp.1565-1568,2007.

[8] Qin, A. Kai, Vicky Ling Huang, and Ponnuthurai N. Suganthan., "Differential evolution algorithm with strategy adaptation for global numerical optimization.", *IEEE transactions on Evolutionary Computation 13,* no. 2 (2009), pp. 398-417,2009.

[9] Takahama, Tetsuyuki, and Setsuko Sakai., "Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation.", *In IEEE congress on evolutionary computation,* IEEE, pp. 1-9, 2010.

[10] Wang, Yong, Zixing Cai, and Qingfu Zhang., "Differential evolution with composite trial vector generation strategies and control parameters.", *IEEE Transactions on Evolutionary Computation 15,* no. 1 (2011), pp. 55-66, 2011.

[11] Qu, Bo-Yang, Ponnuthurai Nagaratnam Suganthan, Jane-Jing Liang, "Differential evolution with neighborhood mutation for multimodal optimization.", *IEEE transactions on evolutionary computation 16,* no. 5 (2012), pp. 601-614,2012.

[12] Elsayed, Saber M., Ruhul A. Sarker, and Daryl L. Essam., "An improved self-adaptive differential evolution algorithm for optimization problems.", *IEEE Transactions on Industrial Informatics 9* , no. 1 (2013), pp.89-99,2013.

[13] Yu, Wei-Jie, Meie Shen, Wei-Neng Chen, Zhi-Hui Zhan, Yue-Jiao Gong, Ying Lin, Ou Liu, and Jun Zhang. ,"Differential evolution with two-level parameter adaptation." *IEEE T. Cybernetics 44*, no. 7 (2014), pp. 1080-1099,2014.

[14] Gong, Wenyin, Zhihua Cai, and Dingwen Liang., "Adaptive ranking mutation operator based differential evolution for constrained optimization.", *IEEE transactions on cybernetics 45*, no. 4 (2015), pp.716-727,2015.

[15] Qiao, Dapeng, and Grantham KH Pang., "A Modified Differential Evolution with Heuristic Algorithm for Nonconvex Optimization on Sensor Network Localization.", *IEEE Transactions on Vehicular Technology 65*, no. 3 (2016), pp. 1676-1689, 2016.

[16] Das S, Abraham A, Chakraborty UK, Konar A. , "Differential evolution using a neighborhood-based mutation operator.", *IEEE Trans Evol Comput* 2009;13(3), pp.526–53,2009.

[17] Das, Swagatam, and Ponnuthurai Nagaratnam Suganthan., "Differential evolution: a survey of the state-of-the-art.", *IEEE transactions on evolutionary computation* 15, no. 1 (2011), pp. 4-31,2011.

[18] Storn, Rainer, and Kenneth Price., "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces." ,*Journal of global optimization 11*, no. 4 (1997), pp. 341-359,1997.

[19] Das S., Suganthan P. N., "Differential evolution: a survey of the state-of-the-art." Evolutionary Computation, *IEEE Transactions on 15.1* ,2011, pp. 4-31,2011.

[20] Rahnamayan, Shahryar, Hamid R. Tizhoosh, and Magdy MA Salama., "Opposition-based differential evolution.", *IEEE Transactions on Evolutionary computation 12,* no. 1 (2008), pp. 64-79,2008.

[21] Ramadas, Meera, and Sushil Kumar, "An efficient hybrid approach using differential evolution and flower pollination algorithm.", *6th International Conference-Cloud System and Big Data Engineering (Confluence),* IEEE, 2016, pp.59-64,2016.

[22] Gandomi, Amir Hossein, Xin-She Yang, Siamak Talatahari, and Suash Deb, "Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization.", *Computers & Mathematics with Applications 63.1* (2012), pp. 191-200, 2012.

[23] Ali, Musrrat, Millie Pant, and Ajith Abraham, "Simplex differential evolution.",*Acta polytechnic Hungarica 6,* no. 5 (2009), pp. 95-115,2009.

[24] Das, Swagatam, Ajith Abraham, And Amit Konar, "Particle Swarm Optimization And Differential Evolution Algorithms: Technical Analysis, Applications And Hybridization Perspectives.", Advances Of Computational Intelligence in Industrial Systems, Springer Berlin Heidelberg, 2008, pp.1-38

[25] Das, Swagatam, Amit Konar, and Uday K. Chakraborty, "Two improved differential evolution schemes for faster global search.", In: *Proceedings of the 7th annual conference on Genetic and evolutionary computation,* ACM, 2005, pp. 991-998, 2005.

[26] Alam, S., Tawseef, M., Khan, F., Fattah, A.A. and Kabir, M.R.," Differential Evolution with Alternating Strategies: A Novel Algorithm for Numeric Function Optimization", *Communications on Applied Electronics (CAE) – ISSN : 2394-4714 Foundation of Computer Science FCS, New York, USA Volume 4– No.2,* January 2016, pp.12-16,2016.