International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM)
ISSN: 2150-7988 Vol.2 (2010), pp.077-086
http://www.mirlabs.org/ijcisim

An Empirical Performance Analysis of Differential Evolution Variants on Unconstrained Global Optimization Problems

G. Jeyakumar,[1] C. Shunmuga Velayutham [2]

Amrita School of Engineering
Amrita Vishwa Vidyapeetham, Coimbatore
Tamil Nadu, India
[1]g_jeyakumar@cb.amrita.edu
[2]cs_velayutham@cb.amrita.edu

Abstract

In this paper we present an empirical, comparative performance, analysis of fourteen Differential Evolution (DE) variants on different classes of unconstrained global optimization benchmark problems. This analysis has been undertaken, with an objective, to compare and to identify competitive DE variants which perform reasonably well on problems with different features. Towards this, fourteen variants of DE are benchmarked on 14 high dimensional unconstrained test functions grouped by their modality and decomposability viz. unimodal separable, unimodal nonseparable, multimodal separable and multimodal nonseparable. The analysis identifies the overall competitive variants as well as the feature based performances of all the variants.

Keywords: Differential Evolution (DE), benchmarking DE variants, empirical performance analysis, unconstrained numerical optimization, success performance

1. Introduction

Differential Evolution (DE), proposed by Storn and Price in 1995 [1], is a simple yet powerful Evolutionary Algorithm (EA) for global optimization in the continuous search domain [2]. The conceptual and algorithmic simplicity, high convergence characteristics and robustness of DE has made it an efficient and popular technique for real-valued parameter optimization [3, 4]. By virtue of these characteristics, DE has shown superior performance in both widely used benchmark functions and real-world problems [5].

Like other EAs, DE is a population-based stochastic global optimizer employing mutation, recombination and selection operators and is capable of solving reliably nonlinear and multimodal problems. However, it has some unique characteristics that make it different from other members of the EA family. The major differences are in the way the offspring (trial vectors) are generated and in the selection mechanism employed to transit to the next generation. In fact, DE uses a differential mutation operation, based on the distribution of parent solutions in the current population, coupled with recombination with a predetermined target vector to generate a trial vector followed by a one-to-one greedy selection between the trial vector and the target vector.

The differential mutation operation essentially perturbs a parent solution with the weighted difference of one/two pair(s) of parent solutions. Depending on the way the parent solutions are selected, both for perturbation and to be perturbed, a number of variants of DE have been proposed in the literature. These variants vary in the type of recombination operator used, the way in which the mutation is implemented and hence consequently in their efficacy to solve a particular problem. Therefore, with the multitude of variants available, choosing the best variant for a given problem is often not easy. So far, no single DE variant has turned out to be best for all problems which is quiet understandable with regard to the No Free Lunch Theorem [6]. Nevertheless, the DE variants need further investigation under which circumstance they perform well. Since, to successfully solve a specific optimization problem at hand, this information can be very important as it may save both time and computational cost.

In this paper, an empirical analysis of the performance of fourteen DE variants on fourteen benchmark problems has been carried out. This analysis has been undertaken to identify the competitive DE variants which perform reasonably well on a range of problems with different features. We understand that testing the DE variants on a limited set of fourteen benchmark problems may not guarantee a complete picture on the relationship between characteristics of the problem and the variant's behavior. Nevertheless, the analysis indeed gives insights about the exploratory efficacy of different variants and hence their utility for solving a range of optimization problems with varying features.

The remainder of the paper is organized as follows. Section 2 describes the DE algorithm and the fourteen variants used for analysis which is followed by a review of related work in Section 3. Section 4 details the design of experiments. Section 5 presents the results obtained over a suite of 14 numerical optimization problems and finally Section 6 concludes the work.

2. DE Algorithm

DE algorithm aims at exploring the search space by sampling at multiple, randomly chosen *NP* *D*-dimensional parameter vectors (population of initial points), so-called individuals, which encode the candidate solution $X_{i,G} = \{x^1_{i,G},\ldots,x^D_{i,G}\}$, $i = 1,\ldots, NP$. The initial population should sufficiently cover the search space as much as possible, by uniformly randomizing individuals, for better exploration. After population initialization an iterative process is started and at each iteration (generation) a new population is produced until a stopping criterion is satisfied.

At each generation, DE employs the differential mutation operation to produce a mutant vector, $V_{i,G} = \{v^1_{i,G},\ldots,v^D_{i,G}\}$ with respect to each individual, $X_{i,G}$ the so called target vector, in the current population. The mutant vector is created using the weighted difference of parent solutions in the current population. A number of differential mutation strategies have been proposed in the literature that primarily differs in the way the mutant vector is created. Along with the strategies came a notation scheme to classify the various DE-variants. The notation is defined by *DE/a/b/c* where '*a*' denotes the base vector or the vector to be perturbed (which can be a random vector, best vector or sum of target vector and weighted vector difference between random and best vectors / random vectors / best vectors); '*b*' denotes the number of vector differences used for perturbation (which can be one or two pairs); and '*c*' denotes the crossover scheme used (which can be binomial or exponential) between the mutant $V_{i,G}$ vector and the target vector $X_{i,G}$ to create a trial vector $U_{i,G} = \{u^1_{i,G},\ldots,u^D_{i,G}\}$.

The seven commonly used mutation strategies are enumerated as follows:

1. DE/rand/1
$$V_{i,G} = X_{r_1^i,G} + F.(X_{r_2^i,G} - X_{r_3^i,G}) \quad (1)$$
2. DE/best/1
$$V_{i,G} = X_{bestG} + F.(X_{r_1^i,G} - X_{r_2^i,G}) \quad (2)$$
3. DE/rand/2
$$V_{i,G} = X_{r_1^i,G} + F.(X_{r_2^i,G} - X_{r_3^i,G} + X_{r_4^i,G} - X_{r_5^i,G}) \quad (3)$$
4. DE/best/2
$$V_{i,G} = X_{bestG} + F.(X_{r_1^i,G} - X_{r_2^i,G} + X_{r_3^i,G} - X_{r_4^i,G}) \quad (4)$$
5. DE/current-to-rand/1
$$V_{i,G} = X_{i,G} + K.(X_{r_3^i,G} - X_{i,G}) + F.(X_{r_1^i,G} - X_{r_2^i,G}) \quad (5)$$
6. DE/current-to-best/1
$$V_{i,G} = X_{i,G} + K.(X_{bestG} - X_{i,G}) + F.(X_{r_1^i,G} - X_{r_2^i,G}) \quad (6)$$
7. DE/rand-to-best/1
$$V_{i,G} = X_{r_3^i,G} + K.(X_{bestG} - X_{i,G}) + F.(X_{r_1^i,G} - X_{r_2^i,G}) \quad (7)$$

where $X_{best,G}$ denotes the best parent vector in the current population, *F* and *K* commonly known as the *scaling factor* or *amplification factor* is a positive real number that controls the rate of evolution of the population. The indices $r_1^i, r_2^i, r_3^i, r_4^i, r_5^i, i \in 1,\ldots, NP$ are randomly generated anew for each mutant vector and are mutually exclusive i.e., $r_1^i \neq r_2^i \neq r_3^i \neq r_4^i \neq r_5^i \neq i$

After the differential mutation strategy, DE then uses a crossover operation in which the mutant vector $V_{i,G}$ mixes with target vector $X_{i,G}$ and generates a trial vector $U_{i,G}$ or offspring. The two frequently used crossover schemes are binomial (uniform) crossover and exponential crossover. The binomial crossover is defined as follows.

$$u^j_{i,G} = \begin{cases} u^j_{i,G} & \text{if } (rand_j[0,1) \le Cr) \vee (j = j_{rand}) \\ x^j_{i,G} & \text{Otherwise} \end{cases} \quad (8)$$

where $C_r \in (0,1)$ (crossover probability) is a user-specified constant and *rand*$_j$*[0,1)* is the j^{th} evaluation of uniform random number generator. $j_{rand} \in 1,\ldots, D$ is a random parameter index, chosen once for each *i* to make sure that at least one parameter is always selected from the mutant vector $V_{i,G}$.

The exponential crossover is defined as follows

$$u^j_{i,G} = \begin{cases} u^j_{i,G} & \text{for } j = \langle n\rangle_D, \langle n+1\rangle_D, \ldots, \langle n+L-1\rangle_D \\ x^j_{i,G} & \text{for all other } j \in [1, D] \end{cases} \quad (9)$$

where the acute brackets $\langle\ \rangle_D$ denote modulo functions with modulus *D*, the starting index *n* is a randomly selected integer in the range [1,D], and the integer *L*, which denotes the number of parameters that are going to be exchanged between the mutant and trial vectors, is drawn from the same range with the probability C_r.

After the mutation and crossover operations, a one-to-one knockout competition between the target vector $X_{i,G}$ and its corresponding trial vector $U_{i,G}$ based on the objective function values decides the survivor, among the two, for the next generation. The greedy selection scheme is defined as follows (assuming a minimization problem without the loss of generality)

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if } f(U_{i,G}) \le f(X_{i,G}) \\ X_{i,G} & \text{Otherwise.} \end{cases} \quad (10)$$

The above 3 steps of differential mutation, crossover, followed by selection marks the end of one DE generation.

These steps are repeated generation after generation until a stopping criterion is satisfied. Figure 1 depicts the algorithmic description of a typical DE.

```
Initialize P_G={X_1,G,...,X_NP,G} with
X_i,G = {x^1_i,G ,..., x^D_i,G}, i = 1,..., NP ;  G=0
Compute {f(X_i,G),...,f(X_NP,G)}
WHILE stopping criterion is not satisfied DO
   /* Mutation */
   FOR i =1 to NP
      Randomly select r^i_1, r^i_2, r^i_3, r^i_4, r^i_5, i ∈ 1,..., NP
      mutually different and different from i,
      Generate a mutant vector V_i,G = {v^1_i,G ,..., v^D_i,G}
      for each target vector X_i,G via one
      of the equations (1) - (7)
    END FOR
   /*Crossover */
   FOR i = 1 to NP
      Generate a trial vector
      U_i,G = {u^1_i,G ,..., u^D_i,G} for each target vector
      X_i,G by employing binomial crossover
      scheme via equation (8) or using
      exponential crossover scheme via
      equation (9).
   END FOR
   /*Selection */
   FOR i = 1 to NP
         IF f(U_i,G) < f(X_i,G) THEN
               X_i,G+1 ← U_i,G
         Else
               X_i,G+1 ← X_i,G
         END IF
    END FOR
    Increment Generation Count G = G+1
END WHILE
```

Figure 1. DE Algorithm

With seven commonly used mutation strategies and two crossover schemes, there are fourteen possible, so called variants of DE viz. *DE/rand/1/bin, DE/rand/1/exp, DE/best/1/bin, DE/best/1/exp, DE/rand/2/bin, DE/rand/2/exp, DE/best/2/bin, DE/best/2/exp, DE/current-to-rand/1/bin, DE/current-to-rand/1/exp, DE/current-to-best/1/bin, DE/current-to-best/1/exp, DE/rand-to-best/1/bin* and *DE/rand-to-best/1/exp.* This paper empirically compares the performance of all these fourteen variants of DE. The DE variants, henceforth in this paper, will be referred without the prefix DE for the sake of brevity.

3. Related Works

The conceptual simplicity and potential of DE has attracted many researchers, who are working on its improvement, resulting in many variants of the algorithm [7, 8, 9 and 10]. Since the performance of classical DE largely rests on the choice of trial vector generation, a thorough understanding of performance difference between the DE variants is crucial. However, little research effort has been devoted to understand and compare the efficacy of existing DE variants to solve problems with different features. This section provides a brief overview of contemporary research efforts in this direction.

Menzura-Montes et. al. [11] empirically compared the performance of eight DE variants on unconstrained optimization problems. Variants with arithmetic recombination, since they are rotationally invariant, were also considered in their work. The study concluded *rand/1/bin, best/1/bin, current-to-rand/1/bin* and *rand/2/dir* as the most competitive variants. However, the potential variants like *best/2/**, *rand-to-best/1/** and *rand/2/** were not considered in their study.

Babu and Munawar [12] compared the performance of ten variants of DE (excluding the *current-to-rand/1/** and *current-to-best/1/** variants of our variants suite) to solve the optimal design problem of shell-and-tube heat exchangers. Their study involved testing an extensive 9680 combinations of key parameters of DE variants and concluded *best/*/** strategies to be better than *rand/*/** strategies with *best/1/** strategies to be the best out of the ten DE variants considered.

A recent study by Qing [13] compared the performance of *rand/1/bin* and *best/1/bin* strategies against their counterpart variants of a Dynamic Differential Evolution (DDE) proposed in [14]. DDE differs from DE by updating its population dynamically while the classical DE updates its population generation by generation. Qing concluded *DDE/best/1/bin* as the competitive variant among the four variants subjected for analysis.

Qin, Huang and Suganthan [15], recently, proposed a self adaptive DE (SaDE) in which both trial vector generation strategies and their associated control parameter values are gradually self-adapted by learning from their past experiences in generating promising solution. Four variants viz. *rand/1/bin, rand-to-best/2/bin, rand/2/bin* and *current-to-rand/1/bin* were considered in the strategy candidate pool. Though this work does not empirically analyze the performance difference between DE variants, it indeed stresses the importance of understanding the efficacy of different trial vector generation strategies during different stages of evolution.

There are other works abound that primarily compares a particular DE variant with its improved or enhanced version

Table 1. Description of the test functions used in our experiments

f_1 – Sphere model $f_{sp}(x) = \sum_{i=1}^{30} x_i^2$ $-100 \le x_i \le 100$	f_2 – Schwefel's Problem 2.22 $f_{sch}(x) = \sum_{i=1}^{30} \lvert x_i \rvert + \prod_{i=1}^{30} \lvert x_i \rvert$ $-10 \le x_i \le 10$
f_3 - Schwefel's Problem 2.21 $f_{sch3}(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le 30\}$ $-100 \le x_i \le 100$	f_4 – Step Function $f_{st}(x) = \sum_{i=1}^{30} (\lfloor x_i + 0.5 \rfloor)^2$ $-100 \le x_i \le 100$
f_5 – Quartic Function with Noise $f_{qf}(x) = \sum_{i=1}^{30} i x_i^4 + random[0,1)$ $-1.28 \le x_i \le 1.28$	f_6 – Schwefel's Problem 1.2 $f_{schDS}(x) = \sum_{i=1}^{30} \left(\sum_{j=1}^{i} x_j \right)^2$ $-100 \le x_i \le 100$
f_7 – Generalized Schwefel's Problem 2.26 $f_{Gsch}(x) = \sum_{i=1}^{30} \left(x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)\right)$ $-500 \le x_i \le 500$	f_8 – Generalized Restrigin's Function $f_{Grf}(x) = \sum_{i=1}^{30} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$ $-5.12 \le x_i \le 5.12$
f_9 – Bohachevsky Functions $f_{bf}(x) = x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7$ $-100 \le x_i \le 100$	f_{10} – Generalized Rosenbrock's Function $f_{rof}(x) = \sum_{i=1}^{30} \left\lvert 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right\rvert$ $-30 \le x_i \le 30$
f_{11} – Ackley's Function $f_{ack}(x) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2} \right) - \exp\left(\frac{1}{30}\sum_{i=1}^{30} \cos(2\pi x_i) \right)$ $-30 \le x_i \le 30$	f_{12} – Generalized Griewank's Function $f_{gri}(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ $-600 \le x_i \le 600$
f_{13} – Generalized Penalized functions $f_{gpf13}(x) = \frac{\pi}{30}\left\{ 10\sin^2(\pi y_1) + \sum_{i=1}^{29} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2 \right\} + \sum_{i=1}^{30} u(x_i, 10, 100, 4)$ $-50 \le x_i \le 50$	
f_{14} – Generalized Penalized functions $f_{gpf14}(x) = 0.1\left\{ \sin^2(\pi 3 x_1) + \sum_{i=1}^{29} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_{i+1})\right] + (x_n - 1)^2 \left[1 + \sin^2(2\pi x_{30})\right] \right\} + \sum_{i=1}^{30} u(x_i, 5, 100, 4)$ $-50 \le x_i \le 50$	

like in [16] to cite an example, or a DE variant with its enhanced versions and Particle Swarm Optimization (PSO) algorithm [5,17] and with simple EAs [5].

4. Design of Experiments

In this paper, we investigate and compare the performance of the fourteen DE variants, identified and listed in Section 2, on a set of benchmark problems with high dimensionality and different features. We have chosen 14 test functions f_1 - f_{14} [11, 18] of dimensionality 30, grouped by the features – unimodal separable, unimodal nonseparable, multimodal separable and multimodal nonseparable. All the chosen test functions are dimensionwise scalable and have an optimum value of zero except f_7.

Table 2. CR values used in our experiments for DE variants

Sno	Variant	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
1	*rand/1/bin*	0.9	0.2	0.9	0.5	0.9	0.2	0.8	0.5	0.1	0.9	0.1	0.1	0.1	0.1
2	*rand/1/exp*	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0	0.9	0.9	0.9	0.9	0.9	0.9
3	*best/1/bin*	0.1	0.1	0.5	0.2	0.8	0.1	0.7	0.1	0.1	0.1	0.1	0.3	0.8	0.1
4	*best/1/exp*	0.9	0.8	0.9	0.9	0.8	0.8	0.9	0.7	0.9	0.8	0.8	0.9	0.8	0.8
5	*rand/2/bin*	0.3	0.1	0.9	0.2	0.9	0.2	0.9	0.2	0.1	0.1	0.1	0.1	0.1	0.1
6	*rand/2/exp*	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.3	0.9	0.9	0.9	0.9	0.9	0.9
7	*best/2/bin*	0.1	0.3	0.7	0.2	0.6	0.1	0.5	0.7	0.1	0.4	0.1	0.1	0.1	0.1
8	*best/2/exp*	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.3	0.9	0.9	0.9	0.9	0.9	0.9
9	*current-to-rand/1/bin*	0.5	0.1	0.9	0.2	0.1	0.1	0.2	0.4	0.1	0.1	0.1	0.2	0.3	0.1
10	*current-to-rand/1/exp*	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.3	0.9	0.9	0.9	0.9	0.9	0.9
11	*current-to-best/1/bin*	0.2	0.1	0.9	0.2	0.1	0.3	0.2	0.8	0.1	0.1	0.2	0.2	0.1	0.1
12	*current-to-best/1/exp*	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.2	0.9	0.9	0.9	0.9	0.9	0.9
13	*rand-to-best/1/bin*	0.1	0.1	0.9	0.4	0.8	0.4	0.8	0.8	0.1	0.9	0.1	0.1	0.2	0.1
14	*rand-to-best/1/exp*	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.4	0.9	0.9	0.9	0.9	0.9	0.9

For the ease of comparison, the description of f7 was adjusted to have its optimum value of zero by adding the optimal value 12569.486618164879 [7].

The test functions are listed Table 1, among which f_1-f_5 are unimodal separable, f_6 is unimodal nonseparable, f_7-f_9 are multimodal separable functions and functions f_{10}-f_{14} are multimodal nonseparable.

The three crucial control parameters of the DE algorithm, irrespective of the fourteen variants, are population size (*NP*), scaling factor (*F*) and crossover rate (*CR*). We fixed the population size *NP* as 60 and maximum number of generations as 3000 (consequently, the maximum number of function evaluations calculate to 1, 80,000). The moderate population size and number of generations were chosen to demonstrate the efficacy of DE variants in solving the chosen problems. The variants will stop before the maximum number of generations is reached only if the tolerance error (which has been fixed as an error value of 1 x 10^{-12}) with respect to the global optimum is obtained. Following [11, 19] we defined a range for the scaling factor, $F \in [0.3, 0.9]$ and this value is generated anew at each generation for all variants. We use the same value of *K* as *F*.

The crossover rate, *CR*, was tuned for each variant-test function combination. Eleven different values for the *CR* viz., {0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0} were tested for each variant-test function combination. For combination of each variant-test_function-*CR* value, 50 independent runs were conducted. Based on the obtained results, a bootstrap test was conducted in order to determine the confidence interval for the mean objective function value. The *CR* value corresponding to the best confidence interval (i.e., 95%) was chosen to be used in our experiments. The fourteen DE variants along with the *CR* values for each test function are presented in Table 2.

As EAs are stochastic in nature 100 independent runs were performed, per variant per test function, by initializing the population for every run with uniform random initialization within the search range.

5. Results and Discussion

The mean and standard deviation of objective function value along with the success rate of each variant–test function combination is presented in Table 3, 4 and 5.

The variants *current-to-rand/1/exp* and *current-to-best/1/exp* have not solved any of the unimodal functions. Similar performance have been shown in the case of multimodal functions as well with poor performance on f_7 (less than or equal to 5% of success rate).

However, their binomial counterparts viz. *current-to-rand/1/bin* and *current-to-best/1/bin*, managed to solve most of the functions except f_8 and f_{10}. It is worth noting that these are the two out of the three functions which solved f_{11} (with 56% of success rate). But their performance on f_7 is similar to that of their exponential counterparts. In the case of unimodal test functions, both variants solved f_1 and f_4 with 100% success rate but failed on f_2, f_3, f_5 and f_6.

The variants *best/1/bin* and *best/1/exp* are the only variants which solved f_7 with a high success rate (above 85%), but failed to solve most of the multimodal functions. The binomial variant performed poorly on f_8 and f_{12} while exponential variant did not solve them at all. In case of unimodal functions, the exponential variant *best/1/exp* solved none but f_6 with 56% success rate. However, the binomial variant solved five out of six variants – f_1 and f_5 with relatively lower success rate and f_2, f_3 and f_6 with relatively higher success rate.

The variants *best/2/bin* and *best/2/exp* (which differ from previous pair by the number of perturbing difference

Table 3. Mean and standard deviation (Std) of objective function values and success rate (SR) for the unimodal functions (**f_1- f_6**)

Variant	f_1			f_2			f_3		
	Mean	**Std**	**SR**	**Mean**	**Std**	**SR**	**Mean**	**Std**	**SR**
rand/1/bin	0.0000	0.0000	100	0.0000	0.0000	100	0.0000	0.0000	100
rand/1/exp	0.0000	0.0000	100	0.0000	0.0000	100	3.7639	2.1111	0
best/1/bin	457.2468	623.9873	3	0.1432	0.5529	40	1.9630	4.9992	79
best/1/exp	583.7935	658.5428	0	4.0507	3.0401	0	37.3642	18.2614	0
rand/2/bin	0.0000	0.0000	100	0.0000	0.0000	100	0.0559	0.0112	0
rand/2/exp	0.0039	0.0049	61	0.0224	0.0060	0	32.9047	5.5725	0
best/2/bin	0.0000	0.0000	100	0.0000	0.0000	100	0.0000	0.0000	100
best/2/exp	0.0000	0.0000	100	0.0000	0.0000	100	0.0474	0.0404	1
current-to-rand/1/bin	0.0000	0.0000	100	0.0183	0.0065	0	3.6843	0.3799	0
current-to-rand/1/exp	24.2932	6.3867	0	44.2244	7.3005	0	57.5220	5.1590	0
current-to-best/1/bin	0.0000	0.0000	100	0.0182	0.0065	0	3.7101	0.4050	0
current-to-best/1/exp	24.3697	8.2160	0	45.0446	5.8030	0	56.6723	5.1574	0
rand-to-best/1/bin	0.0000	0.0000	100	0.0000	0.0000	100	0.0000	0.0000	100
rand-to-best/1/exp	0.0000	0.0000	100	0.0000	0.0000	100	3.3773	1.5910	0
Variant	f_4			f_5			f_6		
	Mean	**Std**	**SR**	**Mean**	**Std**	**SR**	**Mean**	**Std**	**SR**
rand/1/bin	0.0200	0.1400	98	0.0040	0.0049	60	0.0729	0.4216	73
rand/1/exp	0.0000	0.0000	100	0.0157	0.0057	0	0.3121	0.5565	4
best/1/bin	437.2500	663.7670	0	0.0876	0.2676	22	13.2715	99.0423	86
best/1/exp	591.8500	379.3213	0	0.0583	0.0737	0	57.3892	337.3408	58
rand/2/bin	0.0000	0.0000	100	0.0107	0.0032	2	1.6441	2.1216	0
rand/2/exp	0.0000	0.0000	100	0.0473	0.0165	0	269.8565	151.0646	0
best/2/bin	0.0700	0.3258	95	0.0029	0.0059	75	0.0000	0.0000	100
best/2/exp	0.3900	0.6801	69	0.0132	0.0055	2	0.0000	0.0000	100
current-to-rand/1/bin	0.0300	0.2216	100	0.0373	0.0079	0	3210.3641	619.0113	0
current-to-rand/1/exp	43.0700	8.6658	0	0.2683	0.0713	0	3110.8981	1104.1524	0
current-to-best/1/bin	0.0000	0.0000	100	0.0387	0.0096	0	3444.0029	792.0355	0
current-to-best/1/exp	41.9500	9.2631	0	0.2606	0.0618	0	2972.6180	1023.4628	0
rand-to-best/1/bin	0.0000	0.0000	100	0.0040	0.0049	60	0.0691	0.3954	79
rand-to-best/1/exp	0.0000	0.0000	100	0.0149	0.0057	0	0.2048	0.3516	10

vectors) solved most of the multimodal functions except f_{11} and f_8 (in case of exponential variant). It is worth noting that these are the only pair that solved f_{10} while the rest of the variants failed. Interestingly, unlike *best/1/** variants *best/2/** variants performed poorly on f_7. The variants solved all the unimodal functions, but the exponential counterpart performed poorly on f_3 and f_5.

While the variant *rand/1/bin* performed well on unimodal functions with high success rate its exponential counterpart performed poorly on f_6 and failed on f_3 and f_5. With respect to multimodal functions the variants performed well on f_9, f_{12}, f_{13} and f_{14}. While *rand/1/bin* solved f_8 with 100% success rate its exponential counterpart failed, however both variants failed totally on f_{10} and f_{11}. They also have a poor performance on f_7.

The poor performance trend on f_7 continued in the case of *rand/2/bin* and *rand/2/exp* as well. While the former solved most of the multimodal functions with 100% success rate except f_{10} and f_7, the latter performed poorly on f_{10}, failed on f_8 and f_{12} and reasonably well on f_9, f_{11} and f_{14}. In the case of unimodal functions *rand/2/bin* performed very well on f_1, f_2 and f_4, poorly on f_5 and failed on f_3 and f_6. However, the exponential counterpart failed on f_2, f_3, f_5 and f_6. It is worth noting that *rand/2/exp* is one of the three variants to solve the

Table 4. Mean and standard deviation (Std) of objective function values and success rate (SR) for the multimodal functions (***f₇- f₁₂***)

Variant	f_7 Mean	f_7 Std	f_7 SR	f_8 Mean	f_8 Std	f_8 SR	f_9 Mean	f_9 Std	f_9 SR
rand/1/bin	0.13490	0.12770	4	0.00000	0.00000	100	0.00000	0.00000	100
rand/1/exp	0.10280	0.13020	7	47.92500	6.65194	0	0.00000	0.00000	100
best/1/bin	0.00200	0.00632	88	4.33100	3.99821	3	12.93320	15.58621	0
best/1/exp	0.00500	0.01952	85	50.74030	13.02422	0	32.18380	12.24000	0
rand/2/bin	0.22370	0.20552	1	0.00000	0.00000	100	0.00000	0.00000	100
rand/2/exp	0.26680	0.28251	2	101.37520	12.40981	0	0.00930	0.00807	26
best/2/bin	0.17090	0.20154	1	0.69400	1.13478	47	0.11550	0.33019	89
best/2/exp	0.08340	0.17745	17	80.63350	16.49633	0	2.52660	1.45867	7
current-to-rand/1/bin	0.13870	0.13939	2	37.74720	3.99287	0	0.00000	0.00000	100
current-to-rand/1/exp	0.11610	0.11123	3	235.14420	16.16070	0	18.35160	1.97843	0
current-to-best/1/bin	0.18940	0.19713	3	37.03980	4.68652	0	0.00000	0.00000	100
current-to-best/1/exp	0.09800	0.11232	5	232.79620	17.22593	0	18.20830	1.71069	0
rand-to-best/1/bin	0.21780	0.20129	0	0.00000	0.00000	100	0.00000	0.00000	100
rand-to-best/1/exp	0.11570	0.12706	6	48.08910	8.56059	0	0.00000	0.00000	100

Variant	f_{10} Mean	f_{10} Std	f_{10} SR	f_{11} Mean	f_{11} Std	f_{11} SR	f_{12} Mean	f_{12} Std	f_{12} SR
rand/1/bin	21.98680	18.08291	0	0.09000	0.00000	0	0.00000	0.00000	100
rand/1/exp	25.47740	34.90935	0	0.09000	0.00000	0	0.04880	0.09338	68
best/1/bin	585899.88	997086.13	0	3.58330	2.51108	0	3.72490	4.73650	1
best/1/exp	64543.84	170039.70	0	6.09370	1.19750	0	5.90640	2.69510	0
rand/2/bin	19.00760	14.37413	0	0.09000	0.00000	0	0.00000	0.00000	100
rand/2/exp	2741.31730	9859.61	0	0.01040	0.02313	64	0.21020	0.21313	3
best/2/bin	2.31670	9.85087	38	0.09000	0.00000	0	0.00000	0.00000	100
best/2/exp	1.12260	1.87195	29	0.82740	0.64798	0	0.03050	0.06114	44
current-to-rand/1/bin	52.81450	22.26188	0	0.01380	0.02707	56	0.00140	0.00697	96
current-to-rand/1/exp	199243.32	463660.17	0	13.82590	1.12971	0	1.21170	0.05337	0
current-to-best/1/bin	56.91020	32.39015	0	0.01160	0.02419	56	0.00130	0.00734	96
current-to-best/1/exp	119685.68	159936.13	0	13.69280	1.12759	0	1.20760	0.05988	0
rand-to-best/1/bin	17.37140	12.76377	0	0.09000	0.00000	0	0.00000	0.00000	100
rand-to-best/1/exp	24.54490	23.31357	0	0.09000	0.00000	0	0.04920	0.09114	69

function f_{11} with relatively higher success rate than other variants.

Finally the variants *rand-to-best/1/bin* and *rand-to-best/1/exp* failed to solve f_{10}, f_{11} and f_7 in the case of binomial variant and f_8 in the case of exponential variant. The former solved rest of the multimodal functions with 100% success rate. However, the exponential variant performed poorly on f_7 and reasonably well on f_{12}. The binomial variant *rand-to-best/1/bin* performed well on unimodal functions solving most of them with 100% success rate. The variant *rand-to-best/1/exp,* however, failed on f_3 and f_5 and performed poorly on f_6.

Figure 2 compares the overall performances of all fourteen variants on all functions (a-c), unimodal functions (d-f) and multimodal functions (g-i) by plotting empirical distribution of normalized success performance [15, 20]. The success performance (*SP*) has been calculated as follows.

$$SP = \frac{\text{mean}\left(\text{function evalutions for successful runs}\right) * \left(\#\text{ total runs}\right)}{\#\text{ successful runs}} \quad (11)$$

A run is considered *successful* if the global optimum is reached with the given precision, before the maximum number of functions evaluations is reached. The success performances of all fourteen variants on each benchmark function are calculated and are normalized by dividing them by the best *SP* on the respective function.

Table 5. Mean and standard deviation (Std) of objective function values and success rate (SR) for the multimodal functions (***f*****13**- ***f*****14**)

Variant	f_{13}			f_{14}		
	Mean	**Std**	**SR**	**Mean**	**Std**	**SR**
rand/1/bin	0.00000	0.00000	100	0.00000	0.00000	100
rand/1/exp	0.00000	0.00000	100	0.00000	0.00000	100
best/1/bin	15.78	90.67	0	973097.03	2098764.28	0
best/1/exp	131448.66	636890.94	0	154434.94	380096.44	0
rand/2/bin	0.00000	0.00000	100	0.00000	0.00000	100
rand/2/exp	0.00000	0.00000	100	0.00820	0.01086	50
best/2/bin	0.00100	0.01000	99	0.00000	0.00000	100
best/2/exp	0.13760	0.32728	69	0.00350	0.00575	69
current-to-rand/1/bin	0.00050	0.00261	96	0.00000	0.00000	100
current-to-rand/1/exp	10.89420	2.96399	0	24.11140	15.43392	0
current-to-best/1/bin	0.00090	0.00288	91	0.00000	0.00000	100
current-to-best/1/exp	10.37260	2.66337	0	23.03830	7.44047	0
rand-to-best/1/bin	0.00000	0.00000	100	0.00000	0.00000	100
rand-to-best/1/exp	0.00000	0.00000	100	0.00000	0.00000	100

As can be seen from equation (11) small values of *SP* and therefore large values in the empirical distribution graphs are preferable. The first variant that reaches (earlier) the top of the graph will be regarded as the best variant.

For the sake of display, the variants have been plotted in three groups. As can be seen from the first row (a-c) of Figure 2, *rand/1/bin, best/2/bin* and *rand-to-best/1/bin* (closely followed by b*est/2/*exp) have displayed overall superior performance. It is worth noting that none of the variants reached top of the graph because all of them failed to solve atleast one or two multimodal test functions. Three of the four variants identified above rely on the strategy of using the best solution found so far for perturbation. It is known that such strategy perform well while solving unimodal problems. However, despite the fact that the strategies relying on best solution are more likely to get stuck at a local optimum and thereby lead to a premature convergence while solving multimodal problems, the variants did perform well on multimodal test functions as well.

The superior performance of the above four variants is evident in the case of unimodal functions, as can be seen in the second row (d-f) of Figure 2. However, in case of multimodal functions, Figure 2 third row (g-i), while *rand/1/bin*, *best/2/exp*, *rand-to-best/1/bin* and *best/1/bin* reached their maximum performance quicker (as is preferred), variants like *rand/2/bin*, *rand/2/exp, current-to-best/1/bin, current-to-rand/1/bin* and *rand-to-best/1/exp* have also displayed competitive performance. The performances of variants *rand/2/exp, current-to-best/1/bin* and *current-to-rand/1/bin* on unimodal and multimodal test functions are contrary. While on the former the three variants displayed poor performance with respect to the other variants, on the latter they have displayed a very competitive performance. It is worth noting that the Figure 2 do not display the variants *best/1/exp*, *current-to-rand/1/exp* and *current-to-best/1/exp* due to their poor overall performance (and due to the scale involved in the graphs as well). Interestingly, as can be seen from the graphs, the binomial variants have consistently shown a relatively better performance against their exponential counterparts.

6. Conclusion

This paper presented an empirical comparative performance analysis of fourteen classical DE variants on fourteen unconstrained global optimization benchmark problems grouped by their modality and decomposability. The variants *rand/1/bin, best/2/bin, rand-to-best/1/bin* and *best/2/exp* have displayed overall superior performance. This superior performance was due to their dominant success on unimodal functions. However, in the case of multimodal functions, variants like *rand/2/bin*, *rand/2/exp, current-to-best/1/bin, current-to-rand/1/bin* and *rand-to-best/1/exp* have also displayed competitive performance. The variants *best/1/exp*, *current-to-rand/1/exp* and *current-to-best/1/exp* have displayed overall worst performance. It is worth noting that the binomial variants have consistently shown a relatively better performance against their exponential counterparts on both unimodal and multimodal test functions. These observations could be further analyzed focusing on why the variants behave so, and this could be addressed in the light of their exploration and exploitation capabilities.

7. References

[1] R. Storn and K. Price , 1995, “Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces”, Technical Report TR-95-012, ICSI .

[2] R. Storn and K. Price , 1997, “Differential Evolution – A Simple and Efficient Heuristic Strategy for Global Optimization and

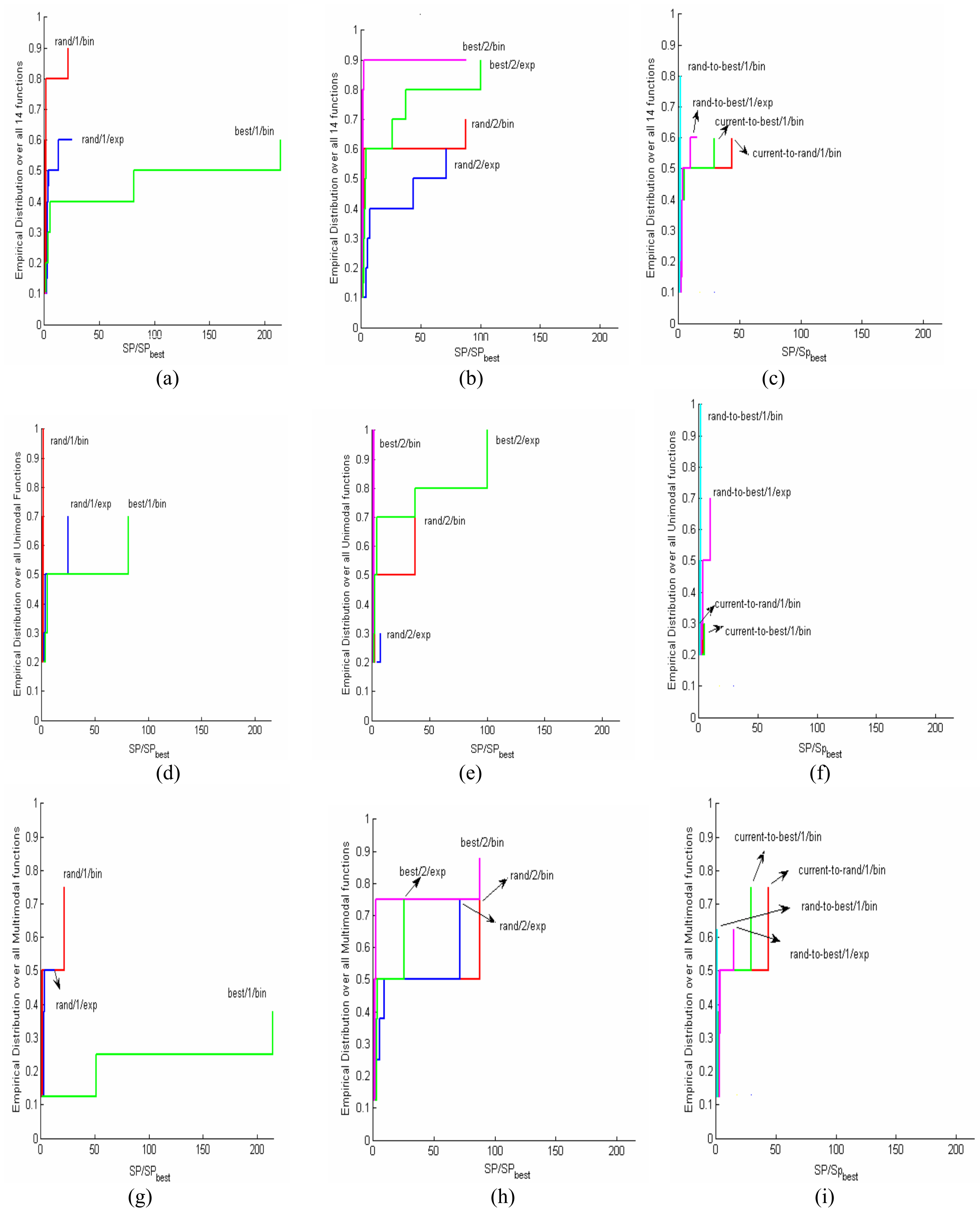

Figure 2. Empirical distribution of normalized success performance of all variants on (a-c) all fourteen test problems, (d-f) unimodal test problems and (g-i) multimodal test problems

Continuous Spaces," Journal of Global Optimization, Vol. 11, No. 4, pp. 341-359.
[3] K .V. Price, 1999, "An Introduction to Differential Evolution," In New Ideas in Optimization, Mc Graw-Hill, UK, D. Corne, M. Dorigo and F. Glover, editors, pp. 79-108.
[4] K. Price, R. M. Storn and J. A. Lampinen, 2005, Differential Evolution : A Practical Approach to Global Optimization, Springer-Verlag, ISBN 3540209506.
[5] J. Vesterstrom and R. Thomsen. June 2004, "A Comparative Study of Differential Evolution, Particle Swarm Optimization and Evolutionary Algorithm on Numerical Benchmark Problems", In Proceedings of the IEEE Congress on Evolutionary Computation (CEC'2004), Vol. 3 , pp. 1980-1987.
[6] David H. Wolpert and William G. Macreedy, 1997, "No Free Lunch Theorems for Optimization", IEEE Transaction on Evolutionary Computation, Vol. 1 No.1 , pp. 67-82.
[7] Ahenya Yang, KeTang and Xin Yao, 2007, "Differential Evolution for High-Dimensional Function Optimization", IEEE Congress on Evolutionary Computation, pp. 3523 – 3530.
[8] M. Pant , M. Ali and A.Abraham , 2009, "Mixed Mutation Strategy Embedded Differential Evolution", Proceedings of the 11th conference on Congress of Evolutionary Computation, Norway, pp. 1240-1246.
[9] M. Pant, R. Thangaraj, A. Abraham and Crime Grosan , 2009, " Differential Evolution with Laplace Mutation Operator", Proceedings of the 11th conference on Congress of Evolutionary Computation, Norway, pp. 2841-2849.
[10] S. Rahnamayan, H. R. Tizhoosh, Magdy and M.A. Salama, 2008, " Opposition-Based DE", IEEE Transaction of Evolutionary Computation, Vol-12, No. 1, pp. 64-67.
[11] Efren Mezura-Montes, Jesus Velazquez-Reyes and Carios A. Coello Coello, July 8-12 2006, "A Comparative Study on Differential Evolution Variants for Global Optimization," Genetic and Evolutionary Computation Conference (GECCO'06), pp. 485-492.
[12] B. V. Babu and S. A. Munawar, 2001, "Optimal Design of Shell-And-Tube Heat Exchanges by Different Strategies of Differential Evolution", Technical Report, PILANI -333 031, Department of Chemical Engineering, Birla Institute of Technology and Science, Rajasthan, India.
[13] Anyong Qing, 2008, "A Study On Base Vector For Differential Evolution", IEEE World Congress on Computational Intelligence, Hong Kong , pp. 550-556
[14] Anyong Qing, 2006, "Dynamic Differential Evolution Strategy and Applications in Electromagnetic Inverse Scattering Problems", IEEE Transactions on Geoscience and Remote Sensing , Vol. 44, No. 1, pp. 116-125.
[15] A.K. Qin, V.L. Huang, and P.N. Suganthan, 009, "Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization", IEEE Transaction on Evolutionary Computing, Vol. 13, No. 12, pp. 397-417.
[16] L.T.Bui, Y.Shan, F. Ai and H.A. Abbass, 2005, "Comparing Two Variants of Differential Evolution in Real Parameter Optimization", Technical Report TR-ALAR 000504009, School of Information Technology and Electrical Engineering, University of New South Wales, Australia.
[17] Swagatam Das, Amit Konar, Uday K.Chakraboty, 2005 , 'Two Improved DE Schemes for Faster Global Search", Genetic and Evolutionary Computation Conference, pp. 991- 998.
[18] X. Yao, Y. Liu, K. H Liang and G. Lin, 2003, "Fast evolutionary algorithms", In Advances in Evolutionary Computing: Theory and Applications, Springer-Verlag, New York, NY, USA. G. Rozenberg, T. Back, and A. Eiben, editors, pp. 45-94.
[19] Efren Mezura-Montes , Personal Communication, Unpublished
[20] N.Hansen, "Compilation of Results on the 2005 CEC Benchmark Function Set", 2006 [online].Available: http://www/ntu.edu.sg/home/epnsugan/index_files/CEC-05/compareresults.pdf

G.Jeyakumar received his B.Sc degree in Mathematics in 1994 and his M.C.A degree (under the faculty of Engineering) in 1998 from Bharathidasan University, Tamil Nadu, India. He is currently an Assistant Professor(Selection Grade) in the department of Computer Science and Engineering, Amrita School of Engineering , Amrita Vishwa Vidyapeetham University, Tamil Nadu, India since 2000. His research interest include evolutionary algorithm, differential evolution, parallelization of differential evolutions and applications of differential evolution.

C. Shunmuga Velayutham received the B.Sc degree in Physics from Manonmaniam Sundaranar University, Tamilnadu, India, in 1998, M.Sc degree in Electronics and Ph.D. degree in Neuro-Fuzzy Systems from Dayalbagh Educational Institute, Uttar Pradesh, India in 2000 and 2005 respectively. Currently, he is an Assistant Professor in the Department of Computer Science & Engineering, Amrita Vishwa Vidyapeetham, Tamilnadu, India since 2005. His research encompasses theoretical investigation and application potential (esp. in computer vision) of evolutionary computation.