Reflected Adaptive Differential Evolution with Two External Archives for Large-Scale Global Optimization

Rashida Adeeb Khanum
Jinnah College for Women
University of Peshawar
Khyber Pakhtunkhwa, Pakistan.
Email: adeeb_maths@yahoo.com

Nasser Tairan
College of Computer Science,
King Khalid University
Abha, Saudi Arabia
Email: nmtairan@kku.edu.sa

Muhammad Asif Jan
Department of Mathematics
Kohat University of Science & Technology
Khyber Pakhtunkhwa, Pakistan
Email: majan@kust.edu.pk

Wali Khan Mashwani
Department of Mathematics
Kohat University of Science & Technology
Khyber Pakhtunkhwa, Pakistan
Email: mashwanigr8@gmail.com

Abdel Salhi
Department of Mathematical Sciences
University of Essex
Colchester, CO4 3SQ, U.K.
Email: as@essex.ac.uk

***Abstract*—JADE is an adaptive scheme of nature inspired algorithm, Differential Evolution (DE). It performed considerably improved on a set of well-studied benchmark test problems. In this paper, we evaluate the performance of new JADE with two external archives to deal with unconstrained continuous large-scale global optimization problems labeled as Reflected Adaptive Differential Evolution with Two External Archives (RJADE/TA). The only archive of JADE stores failed solutions. In contrast, the proposed second archive stores superior solutions at regular intervals of the optimization process to avoid premature convergence towards local optima. The superior solutions which are sent to the archive are reflected by new potential solutions. At the end of the search process, the best solution is selected from the second archive and the current population. The performance of RJADE/TA algorithm is then extensively evaluated on two test beds. At first on 28 latest benchmark functions constructed for the 2013 Congress on Evolutionary Computation special session. Secondly on ten benchmark problems from CEC2010 Special Session and Competition on Large-Scale Global Optimization. Experimental results demonstrated a very competitive performance of the algorithm.**

***Keywords*—Adaptive differential evolution; large scale global optimization; archives.**

I. Introduction

Optimization deals with finding the optimal solution for single or multi-objective functions [1]. An unconstrained single objective optimization problem can be stated as follows:

$$\text{Minimize } f(\mathbf{x}), \tag{1}$$

where $f(\mathbf{x})$ denotes the objective function, and $\mathbf{x} = (x_1, x_2, ..., x_n)^T$ is an n-dimensional real vector.

DE [2] is a most popular bio-inspired scheme for finding the global optimum $\mathbf{x}^*$ of problem (1). The heuristic is essentially an evolutionary one and relies on the usual genetic operators of mutation and crossover. DE is easy to understand and implement, has a few parameters to control, and is robust. There is no doubt that DE is a remarkable optimizer for many optimization problems. However, it has few drawbacks like, stagnation, premature convergence, and loss of diversity. Since it is a global optimizer, so its local search ability is not that good. More details can be found in [3].

To enhance the performance of DE, many modifications to the classic DE have been suggested and various variants of DE are proposed. A novel work is done by Wang et al. [4], in which they utilized orthogonal crossover instead of binomial and exponential crossover. A group of researchers have introduced new variants like opposition based DE [5], centroid dependent initialization ciJADE [6], cluster-based population initialization (CBPI) [7] jDE [8], genDE [9], Individual dependent Mechanism (IDE) [10] etc. Control parameters adaptation and self-adaptation have devised in [11], [12], jDErpo [13] SaDE [14], JADE [15], [16], EPSDE [17], IDE [18], SHADE [19]L-SHADE [20] [21], EWMA-DECrF [22]. Cooperative coevolution have been brought into DE for large scale optimization [23]. Some researchers applied it to problems from the discrete domain [24], [25], while others are taking the advantage of its global searching in the continuous domains [4], [26]–[28].

In another experiment, adaptive variant of DE, the so-called JADE [15], is proposed for numerical optimization. It has shown performance improvement over the state-of-the-art algorithms, jDE [8], SaDE [29] and DE/rand/1/bin [2] according to the reported results in [15] and [30]. However, JADE is not reliable; on some problems. For instance, it finds the global optima in some runs, but it can also be trapped in local optima [30]. To improve the reliability of JADE, in this paper, we introduce two new strategies in JADE and thus propose Reflected Adaptive Differential Evolution with Two External Archives (RJADE/TA).

The rest of this paper is organized as follows. Section II describes the basic DE and JADE algorithm. Section III presents proposed RJADE/TA. Section IV gives the experimental results and finally Section V concludes this paper and discusses future research directions.

II. Differential Evolution and JADE

A. Differential Evolution

The four main schemes of differential evolution (DE) are detailed as follows.

1) Parent Selection: For each member $\mathbf{x}_i$, $i = 1, 2, ..., N_p$, of the current generation G three other members, $\mathbf{x}_{r_1}$, $\mathbf{x}_{r_2}$ and $\mathbf{x}_{r_3}$ are randomly selected, where r_1, r_2 and r_3 are randomly chosen indices such that r_1, r_2 and $r_3 \in \{1, 2, ..., N_p\}$ and $i \neq r_1 \neq r_2 \neq r_3$. Thus, for each individual, $\mathbf{x}_i$, a mating pool of four individuals is formed in which it breeds against three individuals and produces an offspring.

2) Mutation: After selection, mutation is applied to produce a mutant vector $\mathbf{v}_i$, by adding a scaled difference of the two already chosen vectors to the third chosen vector. i.e.,

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}), \tag{2}$$

where $F \in (0, 2)$ [31] is the scaling factor.

3) Crossover: After mutation, the parameters of the parent vector $\mathbf{x}_i$ and mutant vector $\mathbf{v}_i$ are mixed by a crossover operator and a trial member $\mathbf{u}_i$ is generated as follows:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } rand_j(0,1) \leq CR; \\ x_i & \text{otherwise,} \end{cases} \tag{3}$$

where $j \in \{1, 2, ..., n\}$.

4) Survivor Selection: At the end, the trial vector generated in (3) is compared with its parent on the basis of its objective function value. The fittest will propagate to the next generation. i.e.,

$$\mathbf{x}_{i+1} = \begin{cases} \mathbf{u}_i, & \text{if } f(\mathbf{u}_i) \leq f(\mathbf{x}_i); \\ \mathbf{x}_i, & \text{otherwise.} \end{cases} \tag{4}$$

B. JADE

Before presenting the new algorithm, we give the details of the DE's version JADE, upon which the devised algorithm in this paper is based. JADE [15] is an adaptive version of DE. It improves the performance of DE, by implementing a new mutation strategy DE/current-to-p best with/without external archive, and adaptively controlling the parameters F and CR. JADE adopts the crossover and selection scheme of classic DE as described in Equation (3) and Equation (4). DE/current-to-$p$best strategy incorporates not only the best solution information, but also the information of other good solutions. Specifically, any solution from the top $p\%$ population can be randomly selected in DE/current-to-p best to play the role of the single best solution in DE/current-to-best [15]. Where p is the percentage of top good solutions and the default value for it is 5% of N_p. Other suggested values of p are between 5% and 20%, inclusive. JADE modifies classic DE in three aspects.

1) DE/current/to-$p$best strategy: JADE utilizes two mutation strategies, one with external archive, and the other without it. These strategies are the improvement of DE/current-to-best/1 strategy. They can be expressed as follows [15]:

$$\mathbf{v}_i = \mathbf{x}_i + F_i(\mathbf{x}^p_{best} - \mathbf{x}_i) + F_i(\mathbf{x}_{r_1} - \tilde{\mathbf{x}}_{r_2}), \tag{5}$$
$$\mathbf{v}_i = \mathbf{x}_i + F_i(\mathbf{x}^p_{best} - \mathbf{x}_i) + F_i(\mathbf{x}_{r_1} - \mathbf{x}_{r_2}), \tag{6}$$

where $\mathbf{x}^p_{best}$ is a vector chosen randomly from the top $p\%$ individuals and $\mathbf{x}_i$, $\mathbf{x}_{r_1}$ and $\mathbf{x}_{r_2}$ are chosen from the current population P, while $\tilde{\mathbf{x}}_{r_2}$ is chosen randomly from $P \cup A$. Where A denotes the archive of JADE, which records the inferior parent solutions found during the current generation.

2) Control Parameter Adaptation: For each individual $\mathbf{x}_i$, control parameter F_i and the crossover probability, CR_i are generated independently from Cauchy and Normal distributions, respectively as follows [15]:

$$F_i = rand(\mu F, 0.1) \tag{7}$$
$$CR_i = rand(\mu CR, 0.1), \tag{8}$$

where $rand$ is a uniform random number from $[0, 1]$, and μCR and μF are the means of the Normal and Cauchy distributions with standard deviation 0.1. Cauchy distribution is more helpful than the Normal distribution to diversify the mutation factors and thus prevent premature convergence, which often occurs in mutation strategies if the mutation factors are highly concentrated around a certain value [15]. The standard deviation is chosen to be relatively small (0.1) because otherwise the adaptation does not function efficiently; e.g., in the case of an infinite standard deviation, the truncated Normal distribution gets independent of the value of μCR [15]. CR_i and F_i given in Equations (7) and (8) are then truncated to $(0, 1]$ and $[0, 1]$, respectively. Initially, both μF and μCR are set to 0.5 as suggested in [15]. They are expressed as below [15]:

$$\mu F = (1 - c)\mu F + c \cdot mean_L(S_F) \tag{9}$$

$$\mu CR = (1 - c)\mu CR + c \cdot mean_A(S_{CR}). \tag{10}$$

Here $mean_L$ denotes the Lehmer mean, $mean_A$ denotes the arithmetic mean, and S_F is the set of successful F_i's, while S_{CR} is the set of successful CR_i's at generation G. The Lehmer mean is helpful to propagate larger mutation factors, which in turn improves the progress rate. To the contrary, an arithmetic mean of S_F tends to be smaller than the optimal value of the mutation factor and thus it might cause premature convergence at the end. The parameter c in Equations (9) and (10) is a constant which controls the rate of parameter adaptation and is chosen between 0 and 1. The life span of a successful CR_i or F_i is roughly $\frac{1}{c}$ generations; i.e., after $\frac{1}{c}$ generations, the old value of μCR or μF is reduced by a factor of $(1 - c)^{\frac{1}{c}}$, when c is close to zero, if $c = 0$ no parameter adaptation takes place.

3) Optional External Archive: At each generation, the failed parents are sent to the archive. The Euclidian distance of the archive members from the current population is utilized in the mutation operation in order to diversify the population and avert the premature convergence. If the archive size exceeds N_p, some solutions are randomly deleted from it to keep its size equal to N_p.

III. REVIEW

Almost two decades have been passed when DE was proposed in 1995 to cope with non-differentiable, non-convex and non-linear problems defined in the continuous parameter space [32]. Since then, DE and its uncountable and diversified variants have emerged as one of the most competitive and versatile family of the evolutionary computing optimizers and have been prosperously applied to solve numerous real-world problems from diverse discipline of science and technology [33]. Extensive literature on DE is available, which is evident from the recent surveys on DE [34], [32]. However, this section attempts to review some of the relevant methods. The hybridization of DE with local search strategies is a popular area of research among the practitioners. Many hybrid algorithms have shown significant performance improvement.

In [35] Sequential Quadratic Programming (SQP) is merged in DE algorithm. This new hybrid applies the DE algorithm until function evaluations reach 30% of the maximum function evaluations. It then applies SQP for the first time to the best point thus obtained. Afterwards, SQP is applied after each 100 generations to the best solution of the current search. In this work, the population size keeps reducing dynamically and the process terminates with minimum population size.

In another experiment DE is combined with simplex method and this method is know as NSDE [36]. The authors applied nonlinear simplex method with uniform random numbers to initialize DE population. Initially, N_p individuals are generated uniformly and then next N_p are generated from these N_p points by application of Nelder-Mead Simplex (NMS). Now from $2N_p$ population, the fittest N_p are selected as DE's initial population and the rest of DE is unchanged in this algorithm. Their algorithm only modify DE in the population step.

Further, differential evolution algorithm with localization around the best point (DELB) is proposed in [37]. In DELB the initial evolutionary steps are the same as DE except that the mutation scale factor F is chosen from $[-1,-0.4]\cup[0.4,1]$ randomly for each mutant vector, DELB modifies the selection of DE by introducing reflection and contraction. The trial vector is compared with the current best and the parent vector. If the parent is worse than the trial vector it is replaced by a new concentrated or reflected vector. In DELB, the trial vector can be replaced by its parent vector, or reflected vector or contracted vector, while in classic DE only the trial vector replaces the parent.

Inspired by the above techniques, a new variant RJADE/TA of DE family is presented, which records the best individuals of the optimization process at regular intervals. Besides, it utilizes an reflection strategy of local search for replacing the archived solutions. The detail of RJADE/TA is presented in the following section.

IV. PROPOSED REFLECTED ADAPTIVE DIFFERENTIAL EVOLUTION WITH TWO EXTERNAL ARCHIVES

This section proposes a new DE algorithm, RJADE/TA, which modifies JADE in two aspects, first it introduces a second external archive into JADE, which stores superior solutions of the search at regular intervals of the optimization process. Second, these superior solutions are then reflected by new significant/potantial solutions in the current population. RJADE/TA adopts the same crossover and mutation operations as described in JADE [15]. We have done some modification to the Pseudo-code of JADE; this addition can be seen in lines 26 to 31 of Algorithm 1. Further in the last line the best solution is selected from PUA_2, the rest of the code remains the same.

Algorithm 1 Pseudo-code of RJADE/TA

```
1:  Population size = $N_p$; $FES$ = Number of function
    evaluations; $\kappa$ = interval between second archive updates;
2:  Uniformly and randomly sample $N_p$ solutions,
    $\mathbf{x}_{r_1,G}, \mathbf{x}_{r_2,G}, \ldots, \mathbf{x}_{r_{N_p},G}$ from the search space to
    form the initial population $P$;
3:  Initialize the archives $A = \emptyset$; $A_2 = \emptyset$;
4:  Set $\mu CR = 0.5$; $\mu F = 0.5$; $p = 5\%$; $c = 0.1$;
5:  Set $S_{CR} = \emptyset$; $S_F = \emptyset$;
6:  Evaluate these individuals; Set $FES = N_p$;
7:  **while** $FES < n * 10000$ **do**
8:      Generate $CR_i = rand(\mu CR, 0.1)$;
9:      Generate $F_i = rand(\mu F, 0.1)$;
10:     Select $\mathbf{x}^p_{best,G}$ randomly from $100p\%$ population;
11:     Select $\mathbf{x}_{r_1,G} \neq \mathbf{x}_{i,G}$ randomly from $P$;
12:     Select $\tilde{\mathbf{x}}_{r_2,G} \neq \mathbf{x}_{r_2,G}$ randomly from $P \cup A$;
13:     Generate mutant $\mathbf{v}_i = \mathbf{x}_{i,G} + F_i(\mathbf{x}^p_{best,G} - \mathbf{x}_{i,G}) +$
        $F_i(\mathbf{x}_{r_1,G} - \tilde{\mathbf{x}}_{r_2,G})$;
14:     **for** $j = 1$ to $n$ **do**
15:         **if** $j < j_{rand}$ or $rand(0,1) < CR_i$ **then**
16:             $\mathbf{u}_{j,i,G} = \mathbf{v}_{j,i,G}$;
17:         **else**
18:             $\mathbf{x}_{j,i,G} = \mathbf{x}_{j,i,G}$;
19:         **end if**
20:     **end for**
21:     Select the best between $\mathbf{x}_{i,G}$ and $\mathbf{u}_{i,G}$;
22:     **if** $\mathbf{u}_{i,G}$ is better **then**
23:         $\mathbf{x}_{i,G} \rightarrow A$;,$CR_i \rightarrow S_{CR}$, $F_i \rightarrow S_F$;
24:     **end if**
25:     Delete individuals randomly from A if size $A > N_p$;
26:     Update second archive $A_2$ by sending best point of
        the search to it;
27:     **if** $Gen = \kappa$ **then**
28:         $\mathbf{x}_{best,G} \rightarrow A_2$; and reflect it as
29:         Compute the centroid of $P - \mathbf{x}_{best,G}$ as $\mathbf{x}_{c,G} =$
            $\frac{1}{N_p-1}\sum_{i=2}^{N_p} \mathbf{x}_{i,G}$
30:         Generate reflection point as $\mathbf{x}_{r,G} = \mathbf{x}_{c,G} +$
            $1(\mathbf{x}_{c,G} - \mathbf{x}_{best,G})$
31:     **end if**
32:     $\mu CR = (1-c)\cdot\mu CR + c\cdot mean_A(S_{CR})$;
33:     $\mu F = (1-c)\cdot\mu F + c\cdot mean_L(S_F)$;
34: **end while**
35: **Output**: the solution vector with the smallest objective
    function value from $PUA_2$ in the search.
```

A. Best Solution's Reflection

Early convergence of the algorithms may be achieved due to best solution. Thus to avoid premature convergence, stagnation and local optima RJADE/TA reflects the best solution,

$\mathbf{x}_{best,G}$ of the search process and send it to the archive A_2. To implement the reflection mechanism [38] in RJADE/TA, first the center of mass of the current population P except the best solution $\mathbf{x}_{best,G}$ is computed as:

$$\mathbf{x}_{c,G} = \frac{1}{N_p - 1} \sum_{i=2}^{N_p} \mathbf{x}_{i,G} \qquad (11)$$

where $\mathbf{x}_{c,G}$ denotes the center of mass of $N_p - 1$ individuals, since one candidate solution will be archived, this operation can be seen in Algorithm 1 (line 29). Once the center of mass of $N_p - 1$ individuals is calculated, then the best individual $\mathbf{x}_{best,G}$ (the solution with minimum objective value) of P is reflected through the center of mass $\mathbf{x}_{c,G}$ as follows:

$$\mathbf{x}_{r,G} = \mathbf{x}_{c,G} + 1 \cdot (\mathbf{x}_{c,G} - \mathbf{x}_{best,G}). \qquad (12)$$

Where $\mathbf{x}_{r,G}$ is the mirror image or reflection [38] of $\mathbf{x}_{best,G}$ through the centroid $\mathbf{x}_{c,G}$, this newly produced solution is known as reflected solution. The coefficient of reflection is ”1” as suggested in [38].

The reflected solution replaces $\mathbf{x}_{best,G}$ in the population P and the best solution $\mathbf{x}_{best,G}$ by itself is transferred to the second archive A_2.

B. **Second External Archive in RJADE/TA**

When the search procedure reaches its 50% function evaluations the first archive A_2 update is made. After which A_2 is updated at regular interval of generations κ. As mentioned earlier that JADE has archive A, which stores inferior solutions, if the archive size exceeds N_p; some solutions are removed from it. In contrast the proposed second archive A_2 records the best solution of the search after each κ generations. In other words the best solution of the current population, after κ generations is removed from the search procedure and is kept passive in archive A_2 during the optimization. The objective of sending the best solution from the current optimization process is that the best solution information may cause difficulties such as premature convergence due to the resultant reduced population diversity [15]. Best solution some times mislead the search to local optima or stagnation.

The second archive A_2 is initialized as 0 and is updated with a best solution in each κ generations (see Algorithm 1). The interval between two reflections is κ, this is kept 1000 here. If we reflect the best solution at each generation, there will be one extra evaluation at each generation, which may be a wastage of computational energy. Furthermore, if we store best solution at each generation then the best solution of current generation and the previous will be not much different from each other. Which again will be wastage of computation. That is why we selected κ a 1000. There are few differences in A and A_2 which are given below.

1) A_2 stores best solution of the current population, while A records the recently explored inferior solutions.
2) The size of A is kept N_p, if this size exceeds, some solutions are randomly deleted from A, however in the new archive A_2 the size may exceeds N_p. It keeps the record of all best solutions, no solution is removed from it.
3) A_2 records the best solution (only one solution) of the current generation, this may be a parent solution or a child solution. In contrast, A keeps the inferior parents solutions (more than one) only, it does not record inferior child solution.
4) A_2 is initialized as 0 and is updated after κ generations (1000 say). On the other hand A is updated at the end of each generation.
5) The recorded inferior parents of A are later on utilized in mutation. Where in A_2 the stored best solution is reflected with a new solution; which is sent to the current population. Once a solution is kept in A_2, it remains inactive during the optimization. When the search procedures are terminated, then the second archive’s solution contribute towards the selection of optimal solution.

V. Numerical Experiments and Results

A. Experimental Setup

Experimental validations for the proposed RJADE/TA are conducted on a set of 28 new and complex test functions [39] provided by CEC 2013 special session and a 1000 dimensional functions designed for CEC 2010 competition on large scale global optimization problems [40].

B. CEC 2013 Test Suite

In the CEC 2013 test suite, the previously proposed composition functions of CEC 2005 [2] are enhanced and additional test functions are considered for real parameter single objective optimization. Three types of problems are developed:

- Functions 1-5 are unimodal;
- 6-20 are multimodal functions.
- 21-28 are composit functions, which are designed by combining various problems into a complex landscape.

C. Parameter Settings for CEC 2013 Test Suite

We performed our experiments following the guidelines of the CEC2013 competition [39]. For all the problems, the initialization range is $[-100; 100]$. For all of the problems the number of dimensions are $n = 10$ and 30, and the maximum number of objective function evaluations are $10000 \times n$ per run. When the difference between the values of the best solution found and the optimal (known) solution is 10^{-8} or less, the error is set to 0. The population size is set to 100.

D. Results on CEC 2013 functions

The experimental statistics(best, mean, median, worst and standard deviation) obtained by our algorithm in 51 runs, on 28 functions with dimensions $n = 10$ of the CEC 2013 test functions are summarized in Table I. In Table II, the Mean values of function error values($f(x) - f(x^*)$) obtained by RJADE/TA are presented for $n = 10$. These values are compared with state of the art algorithms, jDE, jDEsoo [41] a new version of jDE, SPSRDEMMS [42] and jDErpo [13]. Among these SPSRDEMMS and jDErpo were specially developed for CEC 2013 competition.

In Table II the - shows that the corresponding algorithm loesses against our RJADE/TA algorithm. The + indicates that the particular algorithm wins against our algorithm, and = reveals that both the algorithms performs equivalently. The outstanding performance of RJADE/TA is clearly visible from Table II, where many negative - signs made this fact evident. It is very clear from the Table that our RJADE/TA algorithm performed significantly better than jDE and jDEsoo algorithms on 15 out of 28 functions, on 4 functions both got similar results. On the other hand jDE and jDEsoo showed better performance on only 9 functions. As compared with SPSRDEMMS, our algorithm found better solutions for 16 out of 28 functions and SPSRDEMMS showed good results on 12 functions. Furthermore, jDErpo and RJADE/TA performed better than each other on 12 functions.

Table III shows the comparison of RJADE/TA against jDE, jDEsoo, SPSRDEMMS and jDErpo for $n = 30$. It is interesting to note that the performance of RJADE/TA increased with the increase in dimension. It found better results for 20 out of 28 function against jDE and jDEsoo. jDE only solved 5 out of 28 problems for 30 dimensions, and jDEsoo got good results on 3 out of 28 functions. SPSRDEMMS and jDErpo performed inferior on 16 functions, and superior on 8 functions only, which can be seen from Table III.

Tables II and III showed the comparison of RJADE/TA against each of the particular algorithms. Here we present the overall percentage of all the algorithms, jDE, jDErpo, SPSRDEMMS, jDErpo and RJADE/TA on 30 dimensional problems. Table IV demonstrates that RJADE/TA performance percentage is 50% while jDErpo is 37%, the remaining three algorithms in comparison performed less than or equal to 25%. This percentage validity is even more clearly visible from the bar graph 1. Each bar shows the number of test problems optimized by particular algorithm. The last bar representing RJADE/TA.

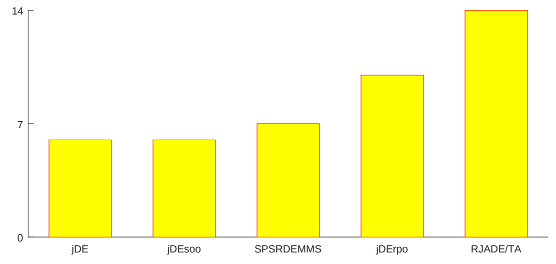

Fig. 1: Comparison of RJADE/TA and other up to date algorithms with dimension $n = 30$

E. CEC2010 Test Instances

Here we evaluate RJADE/TA on ten complex optimization problems used in CEC2010 special session and competition on large scale global optimization [40]. Since separability provides a measure of the complexity of various problems, in [40] a test suite for high dimensional problems is devised which is based on separability and non separability of the functions. Here, three kinds of high-dimensional problems are considered:

- Separable functions;
- Partially-separable functions, in which only a small number of variables are dependent and the rest are independent;
- Partially-separable functions that consist of multiple independent subcomponents, each of which is m-non-separable; and

This test suite provided an enhanced platform for evaluating the performance of algorithms on high-dimensional problems in various scenarios [40]. Below we list only those test functions (F1-F10) which are used in this work.

1) Separable Functions (3)
   - F1: Shifted Elliptic Function
   - F2: Shifted Rastrigin's Function
   - F3: Shifted Ackley's Function
2) Single-group m-nonseparable Functions (5)
   - F4: Single-group Shifted and m-rotated Elliptic Function
   - F5: Single-group Shifted and m-rotated Rastrigin's Function
   - F6: Single-group Shifted and m-rotated Ackley's Function
   - F7: Single-group Shifted m-dimensional Schwefel's Problem 1.2
   - F8: Single-group Shifted m-dimensional Rosenbrock's Function
3) $\frac{n}{2m}$-group m-nonseparable Functions (2)
   - F9: $\frac{n}{2m}$-group Shifted and m-rotated Elliptic Function
   - F10: $\frac{n}{2m}$-group Shifted and m-rotated Rastrigin's Function

The parameter m controls the number of variables in each group and hence defining the degree of separability.

F. Parameter Settings for CEC2010 instances

For this experiment the population size N_p is chosen 50 and the dimension n is set to 1000. The maximum function evaluations are chosen $3 \times 10^{+06}$. The value to reach is set to 10^{-2}. RJADE/TA and JADE were run 25 independent times for all test instances as suggested in the original paper [40]. All these experiments were conducted in MATLAB software.

G. Comparison of RJADE/TA with JADE 0n CEC 2010 instances

The best, median, mean and standard deviation of function error values obtained in 25 runs of the proposed algorithm, RJADE/TA are presented in Table V. These statistics were requested in [40] as well. The best results are typed as bold.

As can be seen from Table V, overall RJADE/TA performed well as compared with JADE in obtaining the "best" solution for five out of ten test instances, F3, F4, F5, F7 and F8. For F6 both algorithms got the same accuracy. Here F3 is separable and all others are single-group m-nonseparable functions. Surly it is due to the additional second archive of RJADE/TA which provides more chance to the population for searching the region and discouraging early convergence. For

TABLE I: EXPERIMENTAL RESULTS OF RJADE/TA ON 28 TEST FUNCTIONS OVER 51 RUNS WITH DIMENSION n = 10.

Func	Best	Worst	Median	Mean	Std Dev
1	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$
2	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$
3	$2.2737E-013$	$9.3924E+02$	$5.1956E+01$	$1.2108E+02$	$1.8941E+02$
4	$0.0000E+00$	$5.9114E+03$	$0.0000E+00$	$1.1591E+02$	$8.2776E+02$
5	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$
6	$0.0000E+00$	$9.8124E+00$	$9.8124E+00$	$7.8884E+00$	$3.9346E+00$
7	$1.9695E-03$	$1.2013E+00$	$7.5908E-02$	$1.5927E-01$	$2.1904E-01$
8	$2.0201E+01$	$2.0511E+01$	$2.0358E+01$	$2.0366E+01$	$6.7627E-02$
9	$3.4033E+00$	$6.0808E+00$	$4.4493E+00$	$4.4593E+00$	$6.0360E-01$
10	$5.6843E-014$	$6.8717E-02$	$3.6112E-02$	$3.5342E-02$	$1.4363E-02$
11	$5.6843E-014$	$1.1937E-012$	$2.2737E-013$	$2.4298E-013$	$1.9922E-013$
12	$3.5441E+00$	$1.1542E+01$	$6.7460E+00$	$6.7571E+00$	$1.6197E+00$
13	$3.9347E+00$	$1.1345E+01$	$7.9523E+00$	$7.7246E+00$	$1.9071E+00$
14	$1.0282E-04$	$1.2604E-01$	$2.1817E-03$	$1.1994E-02$	$2.5730E-02$
15	$3.9803E+02$	$9.2821E+02$	$6.5814E+02$	$6.6660E+02$	$1.2744E+02$
16	$6.2944E-01$	$1.4778E+00$	$1.1505E+00$	$1.1336E+00$	$1.8774E-01$
17	$1.0122E+01$	$1.0122E+01$	$1.0122E+01$	$1.0122E+01$	$4.5729E-06$
18	$1.7593E+01$	$3.1133E+01$	$2.2134E+01$	$2.2715E+01$	$2.8525E+00$
19	$2.9479E-01$	$5.2259E-01$	$4.5204E-01$	$4.4224E-01$	$5.3887E-02$
20	$1.2860E+00$	$3.4877E+00$	$2.5708E+00$	$2.5317E+00$	$3.7190E-01$
21	$2.0000E+02$	$4.0019E+02$	$4.0019E+02$	$3.9627E+02$	$2.8033E+01$
22	$1.9796E-02$	$1.1123E+02$	$1.7431E+01$	$2.7022E+01$	$2.6637E+01$
23	$2.8879E+02$	$1.0544E+03$	$6.9580E+02$	$7.0015E+02$	$1.5859E+02$
24	$1.3524E+02$	$2.1472E+02$	$2.0279E+02$	$2.0217E+02$	$1.2455E+01$
25	$2.0003E+02$	$2.1188E+02$	$2.0091E+02$	$2.0314E+02$	$3.6775E+00$
26	$1.0514E+02$	$2.0002E+02$	$1.1187E+02$	$1.2670E+02$	$3.4574E+01$
27	$3.0001E+02$	$4.0438E+02$	$3.0019E+02$	$3.0351E+02$	$1.6372E+01$
28	$1.0000E+02$	$3.0000E+02$	$3.0000E+02$	$2.8824E+02$	$4.7525E+01$

TABLE II: COMPARISON OF RJADE/TA WITH OTHER ALGORITHMS ON THE MEAN OF THE FUNCTION ERROR VALUES AT EXECUTION TERMINATION OVER 51 RUNS, ON 28 TEST FUNCTIONS WITH n=10.

Func	jDE	jDEsoo	SPSRDEMMS	jDErpo	RJADE/TA
1	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$
2	$7.6534e-05$-	$1.7180e+03$-	$6.8886e+02$-	$0.0000e+00$=	$0.0000e+00$
3	$1.3797e+00$+	$1.6071e+00$+	$5.9735e+00$+	$3.7193e-05$+	$1.2108e+02$
4	$3.6639e-08$+	$1.2429e-01$+	$3.8803e-02$+	$0.0000e+00$+	$1.1591e+02$
5	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$
6	$8.6581e+00$-	$8.4982e+04$-	$8.6580e+00$-	$5.3872e+00$+	$7.8884e+00$
7	$2.7229e-03$+	$9.4791e-01$-	$1.8732e-01$-	$1.6463e-03$+	$1.5927e-01$
8	$2.0351e+01$+	$2.0348e+01$+	$2.0348e+01$+	$2.0343e+01$+	$2.0366e+01$
9	$2.6082e+00$+	$2.7464e+00$+	$2.7311e+00$+	$6.4768e-01$+	$4.4593e+00$
10	$4.5263e-02$-	$7.0960e-02$-	$1.0346e-01$+	$6.4469e-02$-	$3.5342e-02$
11	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$=	$0.0000e+00$
12	$1.2304e+01$-	$6.1144e+00$+	$7.5821e+00$+	$1.3410e+01$-	$7.7246e+00$
13	$1.3409e+01$-	$7.8102e+00$-	$1.1042e+01$-	$1.4381e+01$-	$6.7571e+00$
14	$0.0000e+00$+	$5.0208e-02$-	$8.3273e-02$-	$1.9367e+01$-	$1.1994e-02$
15	$1.1650e+03$-	$8.4017e+02$-	$8.3072e+02$-	$1.1778e+03$-	$6.6660e+02$
16	$1.0715e+00$+	$1.0991e+00$+	$1.1871e+00$-	$1.0598e+00$+	$1.1336e+00$
17	$1.0122e+01$=	$9.9240e+00$+	$1.0127e+01$	$1.0997e+01$-	$1.0122e+01$
18	$3.2862e+01$-	$2.7716e+01$-	$2.2949e+01$-	$3.2577e+01$-	$2.2715e+01$
19	$4.3817e-01$+	$3.1993e-01$+	$3.1854e-01$+	$7.4560e-01$-	$4.4224e-01$
20	$3.0270e+00$-	$2.7178e+00$-	$2.5112e+00$+	$2.5460e+00$-	$2.5317e+00$
21	$3.7272e+02$+	$3.5113e+02$+	$3.9234e+02$-	$3.7272e+02$+	$3.9627e+02$
22	$7.9231e+01$-	$9.1879e+01$-	$6.6219e+01$-	$9.7978e+01$-	$2.7022e+01$
23	$1.1134e+03$-	$8.1116e+02$-	$9.4740e+02$-	$1.1507e+03$+	$7.0015e+02$
24	$2.0580e+02$-	$2.0851e+02$-	$2.0442e+02$-	$1.8865e+02$+	$2.0217e+02$
25	$2.0471e+02$-	$2.0955e+02$-	$2.0473e+02$-	$1.9885e+02$+	$2.0314e+02$
26	$1.8491e+02$-	$1.9301e+02$-	$1.6886e+02$-	$1.1732e+02$+	$1.2670e+02$
27	$4.7470e+02$-	$4.9412e+02$-	$4.7300e+02$-	$3.0000e+02$+	$3.0351e+02$
28	$2.9216e+02$-	$2.8824e+02$=	$2.8431e+02$+	$2.9608e+02$-	$2.8824e+02$
-	15	15	16	12	
+	9	9	9	12	
=	4	4	3	4	

TABLE III: COMPARISON OF RJADE/TA WITH OTHER ALGORITHMS ON THE MEAN OF THE FUNCTION ERROR VALUES AT EXECUTION TERMINATION OVER 51 RUNS, ON 28 TEST FUNCTIONS WITH n=30.

Func	jDE	jDEsoo	SPSRDEMMS	jDErpo	RJADE/TA
1	0.0000e+00=	0.0000e+00=	0.0000e+00=	0.0000e+00=	0.0000e+00
2	1.1925e+06-	1.2914e+05-	1.0157e+05-	2.8378e+04-	7.4009e+03
3	5.6216e+06-	9.8414e+06-	1.0951e+07-	8.5740e+01+	2.4293e+05
4	9.3584e+03-	1.9720e+04-	2.4061e+00+	1.7214e+02+	5.1627e+03
5	0.0000e+00=	1.2606e−08=	0.0000e+00=	0.0000e+00=	0.0000e+00
6	1.4157e+01-	7.9292e+00-	1.7463e+01-	7.5852e+00-	1.0356e+00
7	2.6171e+01-	9.8167e+00-	1.1038e+01-	1.1163e+00+	4.2514e+00
8	2.0934e+01+	2.0946e+01-	2.0950e+01-	2.0940e+01-	2.0937e+01
9	1.8151e+01+	2.0971e+01-	2.4903e+01+	3.0923e+01-	2.7961e+01
10	3.8212e−02-	7.9055e−02-	5.3974e−02-	9.2759e−03+	3.7380e−02
11	3.6609e+01-	0.0000e+00=	0.0000e+00=	3.2858e+01-	0.0000e+00
12	1.7135e+02-	4.2835e+01-	4.2650e+01-	1.7995e+02-	3.6994e+01
13	1.8086e+0-2	7.0750e+01-	7.9763e+01-	1.8151e+02-	5.7309e+01
14	3.0639e+03-	1.3327e+00-	3.2550e+00-	1.5120e+03-	1.1223e+00
15	7.2978e+03-	4.8340e+03-	4.4226e+03-	7.1440e+03-	4.1938e+03
16	2.4646e+00-	2.2791e+00-	2.2801e+00-	2.4687e+00-	2.1305e+00
17	7.8765e+01-	3.0434e+01=	3.0440e+01-	7.3585e+01-	3.0434e+01
18	2.1731e+02-	1.2341e+02-	8.9310e+01+	2.1298e+02-	1.0213e+02
19	7.0078e+00-	1.0956e+00+	1.1639e+00+	7.5022e+00-	2.0825e+00
20	1.2564e+01-	1.1639e+01-	1.1236e+01-	1.2268e+01-	1.0858e+01
21	2.7818e+02+	2.9396e+02-	2.8466e+02+	2.8637e+02+	2.9336e+02
22	3.1346e+03-	5.1621e+01+	7.6606e+01+	1.7779e+03-	1.3131e+02
23	7.2920e+03-	4.6061e+03-	4.7713e+03-	7.2374e+03-	4.2998e+03
24	2.5511e+02-	2.4818e+02-	2.5330e+02-	2.0102e+02+	2.1616e+02
25	2.5213e+02+	2.6037e+02+	2.6408e+02+	2.5354e+02+	2.7921e+02
26	2.0015e+02+	2.5758e+02-	2.0001e+02+	2.0000e+02+	2.2275e+02
27	7.8688e+02-	7.2161e+02-	8.8779e+02-	3.7724e+02-	7.1060e+02
28	3.0000e+02=	3.0000e+02=	3.0000e+02=	3.0000e+02=	3.0000e+02
-	20	20	16	16	
+	5	3	8	8	
=	3	5	4	3	

TABLE IV: %age comparison of RJADE/TA with other algorithms

Optimizer	jDE	jDEsoo	SPSRDEMMS	jDErpo	RJADE/TA
No. of probs. optimized	7	6	6	10	14
%age	25%	21%	21%	36%	**50%**

TABLE V: EXPERIMENTAL RESULTS OF JADE, AND RJADE/TA ON 10 TEST INSTANCES OF 1000 VARIABLES WITH $3 \cdot 10^{+06} FES$.Best, Median, Mean AND the Std Dev OF THE FUNCTION ERROR VALUES OBTAINED OVER 25 RUNS.

Test	Best		Mean		Median		Std Dev	
Instance	RJADE/TA	JADE	RJADE/TA	JADE	RJADE/TA	JADE	RJADE/TA	JADE
F1	3.12E+05	**1.44E+05**	1.26E+06	**1.18E+06**	**9.70E+05**	1.05E+06	**8.07E+05**	9.24E+05
F2	1.44E+03	**1.09E+01**	1.63E+03	**7.28E+01**	1.61E+03	**5.87E+01**	1.30E+02	**4.60E+01**
F3	**7.27E−03**	9.49E−01	**5.47E−01**	1.20E+00	**4.32E−01**	1.20E+00	5.43E−01	**1.47E−01**
F4	**6.29E+10**	9.08E+10	**6.60E+12**	8.12E+12	1.80E+11	**1.56E+11**	**1.03E+13**	1.09E+13
F5	**4.93E+007**	5.65E+007	**6.91E+007**	7.62E+007	**6.71E+007**	7.71E+007	1.52E+007	**1.28E+007**
F6	1.97E+01 ≈	1.97E+01	**1.98E+01**	3.52E+04	1.98E+01 ≈	1.98E+01	**2.80E−02**	1.76E+05
F7	**4.61E+05**	4.58E+05	**1.19E+09**	6.29E+07	7.80E+05	**8.53E+05**	**2.47E+09**	1.36E+08
F8	**7.10E+04**	9.00E+04	4.07E+07	**2.60E+07**	**4.11E+06**	6.97E+06	6.00E+07	**3.49E+07**
F9	3.98E+07	**3.65E+07**	5.03E+07	**4.83E+07**	5.02E+07	**4.51E+07**	**1.04E+07**	7.52E+06
F10	5.04E+03	**3.34E+03**	5.35E+03	**3.67E+03**	5.35E+03	**3.69E+03**	1.66E+02	**1.59E+02**

the remaining four test instances, F1, F3, F9 and F10 JADE got better solutions than RJADE/TA; here F1 and F3 are separable and two functions F9 and F10 are partially-separable functions that consist of multiple independent subcomponents. Furthermore, the failure on F10 ($\frac{n}{2m}$-group nonseparable) could be its complexity, as it is the sum of ten rotated Rastrigins functions applied to groups of m (50 here) decision variables each and one non-rotated Rastrigins function applied to the remaining 500 decision variables. The failure on F9 can be due to its complex nature like F10.

Considering "Mean", "Median" and Standard deviation, we see that RJADE/TA's is more suitable to solve single-group m-nonseparable functions, F3-F8, which is visible from Table V. Hence in general, the analysis of above experimental results lead us to the conclusion that RJADE/TA in much much better than JADE in optimizing problems from the category of single-group m-nonseparable functions.

VI. Conclusions

The current DE variant JADE with one optional external archive some times exhibit poor reliability [30]. Moreover, best solutions some times mislead the search to a local optima. In this paper, we have attempted to introduce a second archive A_2 into JADE for overcoming this shortcoming for large scale global optimization problems. This archive stores the best solution, which is removed from the current population after regular intervals. The removal of best solution is compensated by a new potential solution in the population. Thus we have proposed an approach RJADE/TA to add A_2 to JADE algorithm and add new good divers solutions to the population to make a systematic and rational search in the region defined for the search process. RJADE/TA takes the advantages of both archives, A with inferior solutions and A_2 with superior solutions. It is easy to implement and does not introduce any complicated structures.

The performance of the developed RJADE/TA has been demonstrated by taking advantage of 28 complex competition test functions from CEC 2013 and 10 functions from CEC2010. On CEC2013 test suit RJADE/TA was compared with jDE, jDEsoo, jDErpo and SPSRDEMMS algorithms on 10 and 30 dimensions. The superior performance of RJADE/TA was demonstrated on 10 and 30 dimensions. Moreover, we have compared RJADE/TA with classical JADE with 1000 dimensions. RJADE/TA notably outperformed JADE and is very competitive in solving single-group m-nonseparable functions. In this paper, our aim was to analyze the behavior of algorithm if the best solution is removed from it.

In future JADE with second Archive only can be explored. The experiments may be carried out at other higher dimensional problems. This may be extended to constrained optimization.

References

[1] B. D. Bunday, *Basic optimisation methods*. Arnold, London, 1984.

[2] R. Storn and K. V. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[3] R. Storn and K. Price, "Home page of differential evolution," 2003.

[4] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[5] S. Rahnamayan, T. H. R., and M. M. A. Salama, "Opposition-based differential evolution (ode)," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[6] R. A. Khanum and M. A. Jan, "Centroid-based initialized jade for global optimization," in *Computer Science and Electronic Engineering Conference (CEEC), 2011 3rd*. IEEE, 2011, pp. 115–120.

[7] I. Poikolainen, F. Neri, and F. Caraffini, "Cluster based population initialization for differential evolution frameworks," *Information Sciences*, vol. 297, pp. 216–235, 2015.

[8] J. Brest, , S. Greiner, B. Boskovic, V. Zumer, and M. M. S, "Self-adaptive control parameters in differential evolution: A comparative study on numerical benchmark problems," pp. 446–657, 2006.

[9] N. Noman and H. Iba, "Acceleratinging differential evolution using an adaptive local search," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 3, pp. 107–125, 2008.

[10] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual-dependent mechanism," *Evolutionary Computation, IEEE Transactions on*, vol. 19, no. 4, pp. 560–574, 2015.

[11] J. Brest, A. Zamuda, B. Boskovic, S. Greiner, and V. Zumer, *Advances in Differential Evolution*, 2008, vol. SCI143, ch. An Analysis of the control parameters' Adaptation in Differential Evolution.

[12] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *IEEE Congress on Evolutionary Computation (CEC 2008)*. Singapore: IEEE Press, 2008.

[13] J. Brest, A. Zamuda, I. Fister, and B. Boskovic, "Some improvements of the self-adaptive jde algorithm," in *Differential Evolution (SDE), 2014 IEEE Symposium on*. IEEE, 2014, pp. 1–8.

[14] A. K. Qin and P. N. Suganthan, "Self adaptive differential evolution algorithm for numerical optimization," in *IEEE Congress on Evolutionary Computation (2005)*, vol. 2, 2005, pp. 1785–1791.

[15] J. Zhang and A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 5, pp. 945–958, 2009.

[16] C. Zhang and L. Gao, "An effective improvement of jade for real-parameter optimization," in *Advanced Computational Intelligence ICACI, 2013 Sixth International Conference on*. IEEE, 2013, pp. 58–63.

[17] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing*, vol. 11, pp. 1979–1696, 2010.

[18] M. Ali, M. Pant, and V. P. Singh, "An improved differential evolution algorithm for real parameter optimization problems," *International Journal of Recent Trends in Engineering*, vol. 1, 2009.

[19] R. Tanabe and F. Alex, "Success history based parameter adaptation for differential evolution," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 71–78.

[20] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 1658–1665.

[21] S.-M. Guo, J. S.-H. Tsai, C.-C. Yang, and P.-H. Hsu, "A self-optimization approach for l-shade incorporated with eigenvector-based crossover and successful-parent-selecting framework on cec 2015 benchmark set," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, 2015, pp. 1003–1010.

[22] J. Aalto and J. Lampinen, "A mutation and crossover adaptation mechanism for differential evolution algorithm," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 451–458.

[23] Zheng, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Journal of Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[24] C. Deng, B. Zhao, A. Y. Deng, and C. Liang, "Hybrid-coding binary differential evolution algorithm with application to 0-1 knapsack problems," pp. 317–320, 2008.

[25] C. S. Deng, B. Y. Zhao, and C. Y. Liang, "Hybrid binary differential

evolution algorithm for 0-1 knapsack problem," *Computer Engineering and Design*, vol. 31, no. 8, pp. 1795–1798, 2010.

[26] C. Segura, C. A. C. Coello, E. Segredo, and C. León, "An analysis of the automatic adaptation of the crossover rate in differential evolution," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 459–466.

[27] A. Qin, K. Tang, H. Pan, and S. Xia, "Self-adaptive differential evolution with local search chains for real-parameter single-objective optimization," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 467–474.

[28] F. Wei, Y. Wang, and T. Zong, "Variable grouping based differential evolution using an auxiliary function for large scale global optimization," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 1293–1298.

[29] A. K. Qin, V. L. Huang, and P. N. Sughanthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 398–417, Oct,2009.

[30] F. Peng, K. Tang, G. Chen, and X. Yao, "Multi-start JADE with knowledge transfer for numerical optimization," in *IEEE Congress on Evolutionary Computation (2007)*, 2009.

[31] S. Das and P. N. Suganthan, "Tutorial:differential evolution, foundations, prospectives and applications," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, Paris, France, April, 2011, pp. 1–59.

[32] S. Das, S. S. Mullick, and P. Suganthan, "Recent advances in differential evolution an updated survey," *Swarm and Evolutionary Computation*, pp. 1–62, 2016.

[33] Swagatam and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *Evolutionary Computation, IEEE Transactions on*, no. 99, pp. 1–28, 2011.

[34] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1, pp. 61–106, 2010.

[35] J. Brest, A. Zamuda, B. Bošković, S. Greiner, M. S. Maučce, and V. Žumer, "Self-adaptative differential evolution with sqp local search," in *3rd international conference on Bio-inspired optimization methods and their applications(BIOMA)*, 2008, pp. 59–66.

[36] M. Ali, M. Pant, and A. Abraham, "Simplex differential evolution," *Acta Polytechnica Hungarica*, vol. 6, no. 5, 2009.

[37] P. Kaelo and M. M. Ali, "A numerical study of some modified differential evolution algorithms," *European journal of operational research*, vol. 169, no. 3, pp. 1176–1184, 2006.

[38] J.A.Nelder and R. Mead, "A simplex method for function minimization," *Comput.J*, vol. 7, no. 4, pp. 308–313, 1965.

[39] J. Liang, B. Qu, P. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization," 2013.

[40] K. Tang, Xiodongo, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC2010 special session and competition on large scale global optimization," Nature Inspired Computation and Application Laboratory (NICAL), University of Science and Technology of China, Tech. Rep., 2010.

[41] J. Brest, B. Boskovic, A. Zamuda, I. Fister, and E. Mezura-Montes, "Real parameter single objective optimization using self-adaptive differential evolution algorithm with more strategies," in *Evolutionary Computation (CEC), IEEE Congress on*. IEEE, 2013, pp. 377–383.

[42] A. Zamuda, J. Brest, and E. Mezura-Montes, "Structured population size reduction differential evolution with multiple mutation strategies on cec 2013 real parameter optimization," in *Evolutionary Computation (CEC), IEEE Congress on*. IEEE, 2013, pp. 1925–1931.