

# Improvement of Differential Evolution with Multipopulation-based Ensemble of Mutation Strategies

Besma Hezili<sup>1</sup>, Hichem Talbi<sup>1</sup>

<sup>1</sup> MISC Laboratory, Abdelhamid Mehri University Constantine, Algeria

## Abstract

Due to its advantages, DE has been one of the most used evolutionary algorithms (EA) for solving complex optimization problems. Many DE variants have been proposed to enhance its performance. Differential evolution with a multipopulation-based ensemble of mutation strategies (MPEDE) has been considered as one of the most efficient DE variants. Mutation strategies used in MPEDE are *DE/rand/1*, *current-to-rand/1*, *current-to-pbest/1*. An ameliorated multipopulation-based ensemble DE (AMPEDE) is proposed in the present work where we try to improve the performance of MPEDE by utilizing a new ensemble of mutation strategies and presenting a new mutation strategy called *current-to-mpbest/1*. In this strategy, we are interested in escaping from the local optimum, where individuals are attracted to the center of gravity of the pbest solutions. In our experiments, a comparison of AMPEDE with MPEDE and other algorithms on Black Box Optimization Benchmarking (BBOB) tool has been achieved and the results show that AMPEDE is very competitive and provides an excellent performance in dealing with some optimization problems.

## Keywords

Evolutionary algorithms, Differential evolution, Multipopulation, Ensemble of mutation strategies, Numerical optimization.

## 1. Introduction

Differential evolution (DE) has been introduced initially by Storm and Price [1] to find the global optimum of non-linear, non-convex, multi-modal and non-differentiable functions defined in the continuous parameter space [2]. It is considered as one of the most popular evolutionary algorithms due to its advantages (e.g., simplicity, few control parameters, efficiency in dealing with complex optimization problems, etc.) [3, 4].

DE is a population-based stochastic optimization algorithm, which moves the population toward the global optimum by using mutation, crossover, and selection operations at each generation [5, 6]. Its performance depends on the configuration of mutation strategy and control parameters, such as the population size  $NP$ , the scaling factor  $F$  and the crossover rate  $Cr$  [7].

Many optimization problems exist in the real world [4]. For a given optimization problem, during the search process the most suitable control parameters and the proper mutation strategy


---

Tunisian Algerian Conference on Applied Computing (TACC 2021), December 18–20, 2021, Tabarka, Tunisia

✉ besma.hezili@univ-constantine2.dz (B. Hezili); hichem.talbi@univ-constantine2.dz (H. Talbi)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

may not be the same [4]. Some mutation strategies are applied to improve exploitation while others are used to enhance exploration ability [8]. It is noteworthy that some strategies can achieve a trade-off between exploration and exploitation [9]. Concerning control parameter settings, some can speed up the convergence [9] while others are effective for separable functions [10].

In classical DE, only a single mutation strategy is used and the control parameters are fixed, therefore the performance of DE may vary greatly for different optimization problems [4]. To address this issue, researchers have been more attracted to automatically tune parameters and construct excellent ensemble of mutation strategies [4]. Consequently, many improved DE variants have been proposed such as SaDE [11] (ensemble of two mutation strategies and self-adapted parameters), JADE [12] (to develop an excellent mutation strategy and self-adapted parameters), jDE [13] (self-adaptive tuning parameters), CODE [14] (composition of mutation strategies with their own parameters), MPEDE [7] (ensemble of multiple mutation strategies), MMRDE [4] (multiple mutation strategies based on roulette wheel selection).

In our work, we are interested in MPEDE [4], which is a new variant of DE with a multipopulation-based ensemble of mutation strategies. In MPEDE, the whole population is divided into four sub-populations, three of them have the same size and they are called indicator sub-populations and the fourth one has a different size, it is called reward sub-population. Three mutation strategies are used in MPEDE (*DE/rand/1*, *DE/current-to-rand/1*, and *DE/current-to-pbest/1* with an archive). Each one of them is affected randomly to an indicator sub-population. At each generation, the sub-populations are randomly sampled from the entire population. The reward sub-population is affected to the best mutation strategy after a certain number of generations.

Nevertheless, to the best of our knowledge, MPEDE suffers from the following limitations: 1) the waste of calculation time and the non-stability of the population due to resampling of sub-populations at each generation, 2) the lack of exploitation with the mutation strategies regrouping and 3) the risk of premature convergence to a local optimum due to the attraction to the best current solution or one among the group containing the pbest solutions. To overcome these limitations, we propose a new ensemble of mutation strategies (*DE/rand/1*, *DE/target-to-best/1*, *DE/current-to-mpbest/1* without archive) providing a better balance between exploration and exploitation. To reduce the waste of time and the non-stability of the population, the sub-populations were resampled after a certain number of generations instead of doing it at each generation. To reduce the risk of stagnation at a local minimum, we propose replacing *current-to-pbest/1* by a new mutation strategy: *current-to-mpbest/1*. In this strategy, individuals are attracted to the center of gravity of the pbest solutions instead of being attracted to one of them only.

The paper is organized as follows. In section 2, we explain the basic differential evolution. Section 3 is devoted to research findings in relation with the present work. The proposed approach is described in section 4. Section 5 presents the experiment design and discusses the obtained results in comparison to those of some state-of-the-art algorithms. Finally, section 6 provides a conclusion and some future research directions.

## 2. Basic Differential Evolution (DE)

Differential evolution (DE) belongs to the evolutionary algorithms (EA) family [3]. It provides efficiently outstanding solutions for complex numerical optimization problems [3]. In DE, the evolutionary process consists of four basic steps: initialization, mutation, crossover, and selection [2]. Excluding the initialisation, the remaining steps are repeated until the satisfaction of a given stop criteria, for instance reaching the maximum number of function evaluations [2]. In the remainder of this section, we will present more details about the aforementioned steps.

### 2.1. Initialization

The initial population in DE is randomly generated of  $Np$   $d$ -dimensional real-valued decision vectors [2]. Where  $Np$  is the population size and  $d$  is the dimensionality of problem. A decision vector represents an individual. Each individual is composed of  $d$  decision variables. Each one is defined randomly and uniformly in the range  $[X_{min}, X_{max}]$ . So the initial population can be expressed as shown in equation 1:

$$X_j^i = X_{minj} + rand(0, 1) * (X_{maxj} - X_{minj}) \quad (1)$$

Where  $rand(0, 1)$  is a uniformly distributed random number between 0 and 1 [7].

### 2.2. Mutation

After the initialization step, a donor/mutant vector is created at each generation by using one of the mutation strategies for each parent vector  $X_j^i$ . These strategies include the following:

"DE/current-to-rand/1" [15] :

$$\mathbf{U}_{i,G} = \mathbf{X}_{i,G} + K \cdot (\mathbf{X}_{r_1,G} - \mathbf{X}_{i,G}) + F \cdot (\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}) \quad (2)$$

"DE/rand/1" [16] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot (\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}) \quad (3)$$

"DE/rand/2" [17] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot (\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}) + F \cdot (\mathbf{X}_{r_4,G} - \mathbf{X}_{r_5,G}) \quad (4)$$

"DE/best/1" [16] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{best,G} + F \cdot (\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}) \quad (5)$$

"DE/best/2" [16] :

$$\mathbf{V}_{i,G} = \mathbf{X}_{best,G} + F \cdot (\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}) + F \cdot (\mathbf{X}_{r_3,G} - \mathbf{X}_{r_4,G}) \quad (6)$$

Where the indices  $r1, r2, r3, r4$  and  $r5$  are randomly chosen within the range  $[1, NP]$ . These indices are different from  $i$  [2].  $F$  is a positive integer chosen randomly within the range  $[0,1]$ . It is used to scale the difference vectors.  $\mathbf{X}_{best}$  is the best vector according to its fitness value in generation  $G$  (current generation).

### 2.3. Crossover

After mutation, the crossover step is executed to enhance diversity by generating new trial vectors [4] from the parent vector  $\mathbf{X}_{i,G}$ , and its corresponding mutant vector  $\mathbf{V}_{i,G}$ . Two types of crossover operation exist: the binomial crossover and the exponential crossover [2]. The binomial crossover is generally used in DE and it is defined as follows:

$$\mathbf{U}_{i,G}^j = \begin{cases} \mathbf{V}_{i,G}^j & \text{if } (\text{rand}_j[0, 1] \leq Cr) \text{ or } (j = j_{rand}), j = 1, 2, \dots, D \\ \mathbf{X}_{i,G}^j & \text{otherwise} \end{cases} \quad (7)$$

Where  $Cr$  is a control parameter called crossover rate.  $i = 1, 2, \dots, NP$ ,  $j = 1, 2, \dots, D$  where  $NP$  is the population size and  $D$  is the dimensionality of problem.  $j=j_{rand}$  is used to ensure that  $\mathbf{U}_{i,G}$  is different from the target vector  $\mathbf{X}_{i,G}$  at least in one individual.

### 2.4. Selection

According to the fitness value, the selection step chooses between the parent vector and the trial vector the one which will survive in the next generation. It is defined as shown in equation 8

$$\mathbf{X}_{i,G+1} = \begin{cases} \mathbf{U}_{i,G} & \text{if } f(\mathbf{U}_{i,G}) \leq f(\mathbf{X}_{i,G}) \\ \mathbf{X}_{i,G} & \end{cases} \quad (8)$$

## 3. Related Works

DE can be considered as one of the best evolutionary algorithms (EA) [4], mainly because of its simplicity and robustness [7]. Nevertheless, despite these advantages, it suffers from stagnation, premature convergence, and long calculation time [4]. The performance of DE depends on the used mutation strategies and control parameters [18] and choosing suitable strategies and parameters is far from being an easy task. It requires much expertise [19, 20].

Many studies have been proposed to determine the appropriate setting of the control parameters of DE (e.g., size  $NP$ , Crossover  $Cr$ , Scale factor  $F$ , etc.) based on the properties of the problem [7]. In DE, three main control parameters are used:  $NP$ ,  $F$  and  $Cr$ .  $NP$  (population size) has a big effect on the convergence speed of DE. For separable and uni-modal functions, we have to choose a smaller value of  $NP$  to speed up the convergence [19]. While in multi-modal function a larger value of  $NP$  is recommended to avoid premature convergence.  $F$  is a positive control parameter used to scale the difference between vectors. We have to choose carefully the initial value of  $F$ . In [21], 0.9 is claimed to be the best value. Therefore, typical values of  $F$  are generated randomly in the range [0.4, 0.95] [21].  $Cr$  is called crossover rate (crossover probability). It is used to control the ratio of genes from the mutant vectors which will participate in the crossover step. A good choice of  $Cr$  is within the range [0.3, 0.9], while a good initial value is 0.1 [22].

As we have seen in the aforementioned studies, many conclusions were proposed for the manual tuning of control parameters. Nevertheless, during the evolution process of DE, it is worthy to adapt the parameters. In fact, the control parameters can be partitioned into

three categories: deterministic, adaptive, and self-adaptive [23]. In adaptive parameter, a fuzzy adaptive differential evolution (FADE) has been introduced by Liu and Lampinen where  $F$  and  $Cr$  are dynamically adapted [24]. For multiobjective optimization, Zaharie and Petcu have designed an adaptive Pareto DE algorithm and analyzed its parallel implementation [25]. Concerning self-adaptive parameters, Omran et al. proposed (SDE), where a normal distributed  $N(0.5,0.15)$  is used to generate the  $Cr$  for each individual [26]. Brest et al. [27] proposed the jDE which considered as new variant adaptive of DE.  $F$  and  $Cr$  are adaptive during the execution of DE.

The used mutation strategy has a big influence on the performance of DE [4]. Thus, to enhance DE performance, some works are interested in introducing a new mutation strategy [28]. For instance, a new variant of DE/current-to-best, called DE/current-to-pbest, with an optional external archive is proposed by Zhang et al [12], where  $p \in (0, 1]$  and any of the top  $100 \cdot p\%$  individuals can be chosen to be the best. DE/current-to-pbest offers a good balance between exploration and exploitation [7]. DE/current-to-gr-best/1 [29] is also a variant of DE/current-to-best/1 in which the best solution is chosen from a group that contains  $q\%$  of the population (randomly chosen).

Instead of spearing the control parameter tuning and the choice of the suitable mutation strategy, researchers proposed to incorporate ensemble strategies and parameters into evolutionary algorithms [7]. Qin et al. [17] proposed a self-adaptive DE algorithm (SaDE) in which mutant vector generation strategies and their own control parameters are self adapted based on their previous experiences in generating promising solutions. A hybrid mutation strategy have been proposed by Yi et al. [30] as a new variant of DE. It uses two types of mutation strategies and self-adapting control parameters. To solve constrained optimization problems, Wang and co-workers [31] proposed ICDE which uses multiple mutation strategies and the binomial crossover to generate the trial vectors.

## 4. The Proposed Approach

### 4.1. Multipopulation-based ensemble DE (MPEDE)

MPEDE [7] is a variant of DE based on multipopulation. It uses multiple strategies and partitions the whole population into four sub-populations. Three mutation strategies are used in MPEDE  $DE/rand/1$ ,  $DE/current-to-rand/1$ ,  $DE/current-to-pbest$  with archive. Besides that, there are three equal and small-sized sub-populations (indicator sub-populations) and a large-sized sub-population (reward sub-population). After a given number of generations, the best performing mutation strategy is allocated to the reward sub-population and it can win more computational resources. Let  $pop$  determine the entire population and  $pop_j$  represent the  $j$ th sub-population. we can define  $pop$  as:

$$pop = \bigcup_{j=1, \dots, 4} pop_j \quad (9)$$

Let  $NP$  be the size of  $pop$ . The size of  $pop_j$  is  $NP_j$  and it is calculated as follows:

$$NP_j = \lambda_j \cdot NP \quad j = 1, \dots, 4 \quad (10)$$

$$\sum_{j=1,\dots,4} \lambda_j = 1 \quad (11)$$

$\lambda_j$  is the portion between  $pop_j$  and  $pop$ , and  $\lambda_1 = \lambda_2 = \lambda_3$ .

As we have seen before, after a certain number of generations the best performing mutation strategy is rewarded by more computational resource ( $pop_4$ ). The performance of each mutation strategy is calculated by the proportion of fitness improvements ( $\Delta f_j$ ) and consumed function evaluations during the last  $ng$  generations ( $ng.NP_j$ ). The index of the best mutation strategy can be expressed as shown in equation 12 :

$$k = arg \left( max_{1 < j \leq 3} \left( \frac{\Delta f_j}{ng.NP_j} \right) \right) \quad (12)$$

#### 4.2. The Limitations of MPEDE and the Adopted Ensemble of Mutation Strategies

As described above, in MPEDE sub-populations are randomly resampled from the entire population at each generation, This can lead to a loss of computation time. Three mutation strategies are used in MPEDE. First *DE/rand/1* which is considered as the most commonly used mutation strategy. It has a big exploration capacity [11]. Second *current-to-rand/1*, it is very effective in solving multiobjective optimization problems. It is considered as a rotation-invariant strategy [7]. Third *current-to-pbest /1* which is very useful in solving complex optimization problems . With this strategy, problems like premature convergence can be solved due to its ability to diversify the population [12]. *Current-to-rand/1* is used without crossover while *DE/rand/1* and *current-to-pbest* are used with the combination of binomial crossover [7]. We find there is a lack of exploitation in this ensemble of mutation strategies.

In this paper, we propose an improved variant of MPEDE (AMPEDE) in which we try to overcome the limitations that we find, and to enhance MPEDE results. In AMPEDE we divided the entire population into three large equally-sized sub-populations (indicator sub-population) and one small-sized sub-population (reward sub-population). Three mutation strategies are used in our approach. *DE/rand/1* to ensure a good exploration ability [11]. *DE/target-to-best/1* [15] to promote exploitation in our ensemble of mutation strategies. *Current-to-mpbest/1* without archive is used not to be trapped into a local optimum. In this strategy instead of choosing  $\mathbf{X}_{best}$  randomly one of the top 100. $p$ % individuals in the current population with  $p \in (0, 1]$ , we take  $\mathbf{X}_{best}$  from the mean of the top 100. $p$ % individuals in the current population with  $p \in (0, 1]$ . The three mutation strategies are used with the binomial crossover. As in MPEDE [7], after every certain number of generations, the best mutation strategy performing is determined using the ratios of fitness improvements and consumed function evaluations during the previous  $ng$  generations. According to our experimental results, as illustrated in the next section, we find that it is not useful to resample sub-populations from the entire population at each generation. Therefore, in our approach we choose resampling sub-populations after every  $ng$  generations to enhance the stability and the adaptability of the algorithm.

### 4.3. Parameter Adaptation

Because parameters during the evolution process of DE cannot be the same, we need to choose the appropriate control parameters for different mutation strategies to enhance DE performance [8]. Studies like [22, 11, 12] have proposed different approaches to parameter adaptation. In our work, we used the technique defined in [12], which is the same as that used in MPEDE. A Cauchy distribution is used to generate the scaling factor  $F_{i,j}$  of individual  $\mathbf{X}_i$  that uses the  $j$ th mutation strategy. The scaling factor formula is shown in equation 13

$$F_{i,j} = \text{randc}_{i,j}(\mu F_j, 0.1) \quad (13)$$

Where in this Cauchy distribution,  $\mu F_j$  represents the location parameters, and the scale parameter is 0.1. After the update, if  $F_{i,j}$  is greater than 1,  $F_{i,j}$  will be truncated to 1, and it will be regenerated to a feasible value if it is smaller than 0. The initial value of  $\mu F_j$  is set to 0.5 and it is updated as follows:

$$\mu F_j = (1 - c) \cdot \mu F_j + c \cdot \text{mean}_L(S_{F,j}) \quad (14)$$

Where  $S_{F,j}$  is the set of  $F_{i,j}$  used by the  $j$ th mutation strategy and assists this strategy to generate better solutions at generation  $g$ .  $\text{mean}_L$  is the Lehmer mean; it is calculated as below:

$$\text{mean}_L = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \quad (15)$$

The crossover probability  $CR_{i,j}$  of individual  $\mathbf{X}_i$  uses the mutation strategy  $j$  is generated according to a normal distribution. The crossover probability equation is defined as follows:

$$CR_{i,j} = \text{randn}_{i,j}(\mu CR_j, 0.1) \quad (16)$$

Where in this normal distribution  $\mu CR_j$  represents the mean value and 0.1 is the standard deviation value. Initial value of  $\mu CR_j$  is 0.5 and it is updated after each generation as:

$$\mu CR_j = (1 - c) \cdot \mu CR_j + c \cdot \text{mean}_A(S_{CR,j}) \quad (17)$$

Where  $S_{CR,j}$  is the set of  $CR_{i,j}$  used by the  $j$ th mutation strategy and assists this strategy to generate better solutions at generation  $g$ .  $c$  is a positive constant within the range [0,1], and  $\text{mean}_A$  is the arithmetic mean value of elements in the collection  $S_{CR,j}$ . The schema of AMPEDE is given in Algorithm1.

## 5. Experiment and Result Analysis

Our proposed algorithm is tested on the Comparing Continuous Optimizer [32] (COCO) platform. This platform is used for the BBOB workshops. It uses 24 noiseless test functions for testing real parameters optimization problems with single objective functions. According to their features, the functions are partitioned into 5 sub-groups. The BBOB functions are shown in .

**Algorithm 1:** pseudo code of AMPEDE

---

```

1 Set  $\mu CR_j = 1.0$ ,  $\mu F_j = 0.5$ ,  $\Delta f_j = 0$  and  $\Delta f_{es_j} = 0$  for each  $j = 1, \dots, 4$ ;
2 Initialize, NP, ng, for each  $j = 1, \dots, 4$ ;
3 Initialize, the pop randomly distributed in the solution space;
4 Initial  $\lambda_j$  and set,  $NP_j = \lambda_j \cdot NP$ ;
5 Randomly partition pop into  $pop_1, pop_2, pop_3$  and  $pop_4$  with respect to their sizes.;
6 Randomly select a sub-population  $pop_j$  ( $j = 1, 2, 3$ ) and combine  $pop_j$  with  $pop_4$ . Let
    $pop_j = pop_j \cup pop_4$  and  $NP_j = NP_j + NP_4$ ;
7 Set  $g = 0$ ;
8 while  $g \rightarrow MaxG$  do
9    $g = g + 1$ ;
10  for  $j=1 \rightarrow 3$  do
11    Calculate  $\mu CR_j$  and  $\mu F_j$ ;
12    Calculate  $CR_{i,j}$  and  $F_{i,j}$  for each individual  $X_i$  in  $pop_j$ ;
13    Perform the  $j$ th mutation strategy and related crossover operators over
       subpopulation  $pop_j$ ;
14    Set  $SCR_{,j} = \emptyset$   $SF_{,j} = \emptyset$ ;
15  for  $i=1 \rightarrow NP$  do
16    if  $f(X_{i,g}) \leq f(u_{i,g})$  then
17       $X_{i+1,g} = X_{i,g}$ ;
18    else
19       $X_{i+1,g} = u_{i,g}$ ;
20       $CR_{i,j} \rightarrow SCR_{,j}$ ;
21       $F_{i,j} \rightarrow SF_{,j}$ ;
22   $pop = \bigcup_{j=1..3} pop_j$ ;
23  if  $mod(g, ng) = 0$  then
24     $k = arg \left( max_{1 < j \leq 3} \left( \frac{\Delta f_j}{ng \cdot NP_j} \right) \right)$ ;  $\Delta f_j = 0$ ;
25    Randomly partition pop into  $pop_1, pop_2, pop_3$  and  $pop_4$ ;
26    Let  $pop_k = pop_k \cup pop_4$  and  $NP_k = NP_k + pop_4$ ;

```

---

**5.1. Experiment Design**

In the experiments, we compared our proposed approach (AMPEDE) to DE-PSO, GA, JADE, CMAES, and MPEDE. Each of the algorithms was run on 15 instances of all the 24 functions in dimensions 5, 10, 20. The evaluation budget was set to  $10^4 \cdot d$  function evaluations for each run. Feasible solutions during the execution are within  $[-5, 5]$ . Running algorithms is continued until a stop criterion is satisfied: reaching the maximum number of function evaluations or getting a solution close to the best known solution of the problem with a precision greater than  $10^{-8}$ . Many variants with different parameters values are tested in our experiments. The parameters values used by the best variant are: population size  $NP = 150$ , generation gap  $ng = 30$  which is



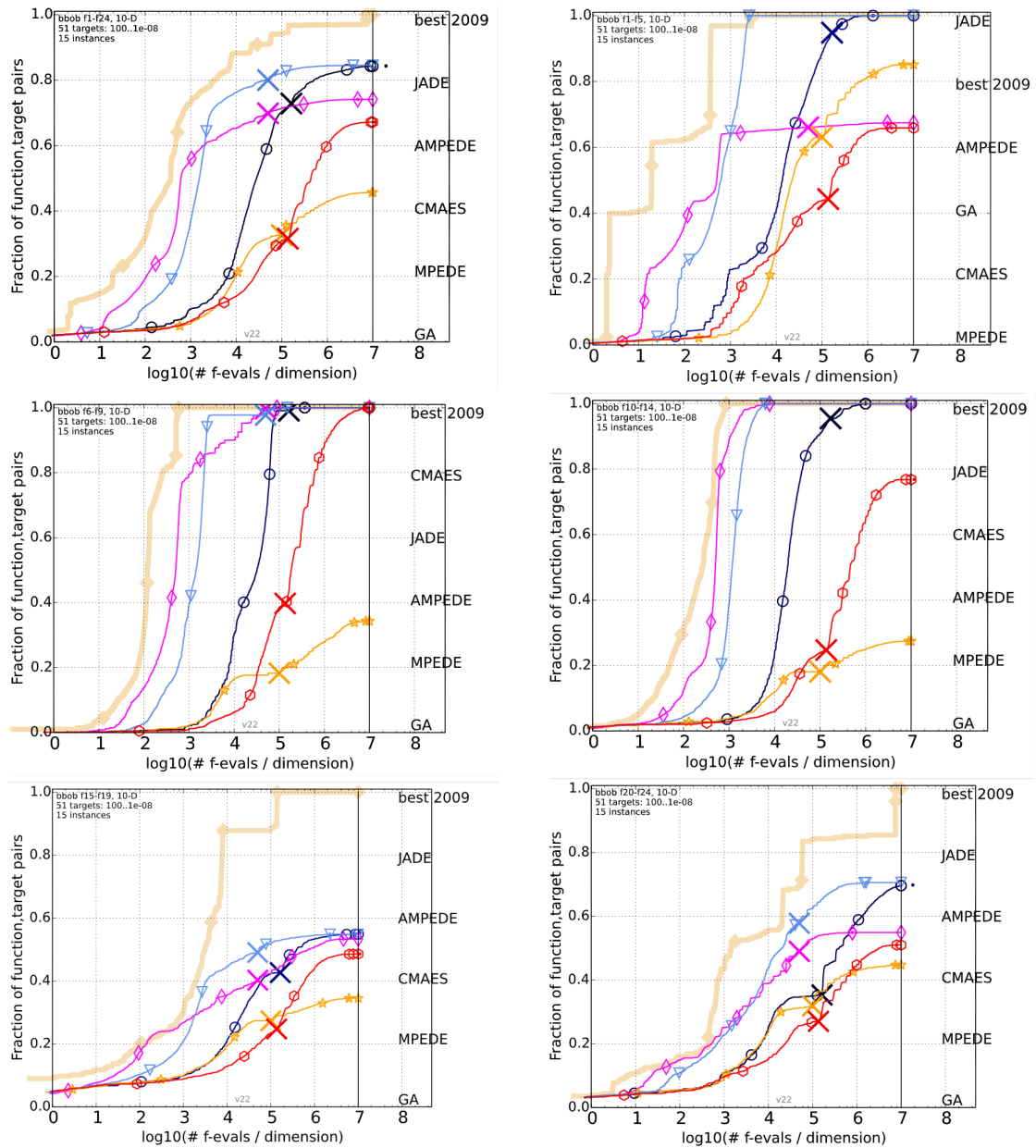
1 Separable Functions	f1: Sphere function f2: Ellipsoidal Function f3: Rastrigin Function f4: Buche-Rastrigin Function f5: Linear Slope f6: Attractive Sector Function
2 Low or moderate conditioning	f7: Step Ellipsoidal Function f8: Rosenbrock Function original f9: Rosenbrock Function rotated f10: Ellipsoidal Function f11: Discus Function
3 Unimodal with high conditioning	f12: Bent Cigar Function f13: Sharp Bridge Function f14: Different Power Function f15: Rastrigin Function f16: Weierstrass Function
4 Adequate global structure with Multi-modal	f17: Schaffers F7 function f18: Schaffers F7 Functions moderately ill-conditioned f19: Composite Griewank-RosenbrockFunction F8F2 f20: Schwefel Function f21: Gallagher's Guassian 101-me PeaksFunction
5 Multi-modal function with weakglobal structure	f22: Gallagher's Guassian 21-hi PeaksFunction f23: Katsuura Function f24: Lunacek bi-Rastrigin Function

**Table 1**  
**Current BBOB Functions**

used to specify the best mutation strategy periodically, proportion between indicator population and entire population  $\lambda_1$  (as  $\lambda_1=\lambda_2=\lambda_3=0.3$ , initial value of  $\mu CR_j=1.0$  and  $\mu F_j=0.5$ . The value of  $p$  in the “*Current-to-mpbest/l*” is 0.1.

## 5.2. Result Analysis

Our experimental results are presented in Figure1 and Figure2. Despite the global superiority of CMA-ES and JADE, results obtained show that AMPEDE has a good performance in separable functions, functions with low or moderate conditioning, functions with high conditioning and unimodal, especially f19 where AMPEDE shows better convergence rate and outperformed all algorithms like GA, JADE, CMAES, MPEDE in 5D. According to the results presented in Figure 1 and Figure2, the changes we made in MPEDE were very useful. As we see AMPEDE has an excellent performance compared to MPEDE in all the benchmark function in 5D and 10D. Despite that AMPEDE suffers in dealing with Multi-modal functions with adequate global structure like f15,f16,f19 in 10D and the Multi-modal functions with weak global structure like f23 and f24.



**Figure 1:** Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/DIM) for 51 targets with target precision in  $10^{[-8]}$  for all functions and subgroups in different dimensions. As reference algorithm, the best algorithm from BBOB 2009

## 6. Conclusion

In this paper, we have presented a new DE variant, named amelioration multipopulation ensemble DE (AMPEDE). AMPEDE is an improvement of MPEDE where we have introduced a new ensemble of mutation strategies instead of the grouping of mutation strategies in MPEDE

$\Delta f_{opt}$	1e1	1e0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
$\Pi, S-D$	11	12	12	12	12	12	12	15/15
AMPEDE	4.8 (8)	604 (263)	1438 (521)	2392 (291)	3341 (460)	5310 (1008)	7254 (462)	15/15
CMAES	<b>2.3 (2)</b>	<b>8.7 (2)*<sup>3</sup></b>	<b>15 (4)*<sup>4</sup></b>	<b>22 (4)*<sup>4</sup></b>	<b>28 (3)*<sup>4</sup></b>	<b>41 (4)*<sup>4</sup></b>	<b>53 (2)*<sup>4</sup></b>	15/15
GA	8.7 (5)	369 (192)	1202 (170)	2130 (405)	2989 (360)	5474 (402)	8468 (548)	13/15
JADE	4.1 (3)	18 (6)	37 (9)	57 (9)	78 (10)	117 (9)	157 (14)	15/15
MPEDE	9.0 (13)	1673 (1816)	1.3e4 (2e4)	3.2e4 (4e4)	5.1e4 (4e4)	8.4e4 (1e5)	1.4e5 (2e5)	5/15

**Figure 2:** Expected running time (ERT in number of function evaluations) divided by the respective best ERT measured during BBOB-2009 (given in the respective first row) for different  $\Delta f$  values in dimension 5. The central 90% range divided by two is given in braces. The median number of conducted function evaluations is additionally given in italics, if  $ERT(10^{-7}) = \infty$ . # succ is the number of trials that reached the final target  $fopt + 10^{-8}$ . Best results are printed in bold

and proposed a new mutation strategy.

Mutation strategies used in AMPEDE are first *DE/rand/1*, second *target-to-rand/1*, third *current-to-mpbest* which is a new mutation strategy that we have proposed. In *current-to-mpbest* individuals are attracted to the center of gravity of the *pbest* solutions instead of being attracted to the a solution chosen randomly from the *pbest* solutions. Based on the results of our experiments, *current-to-mpbest* outperforms *current-to-pbest*.

IMPEDE is compared to MPEDE and others algorithms on BBOB. The experimental results show that AMPEDE provides an obvious performance improvement in comparison to the original MPEDE especially in 2D, 3D, 5D and 20D.

In our future work, we plan to add an adaptive strategy to AMPEDE to enhance its performance and to solve more optimization problems. In addition, we intend to combine AMPEDE with other existing metaheuristics to improve our results.

## Acknowledgments

This work was partially supported by the LABEX-TA project MeFoGL: "Méthodes Formelles pour le Génie Logiciel".

## References

- [1] R. Storn, K. V. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, *J. Glob. Optim.* 11 (1997) 341–359. URL: <https://doi.org/10.1023/A:1008202821328>. doi:10.1023/A:1008202821328.
- [2] S. Das, S. S. Mullick, P. N. Suganthan, Recent advances in differential evolution - an updated survey, *Swarm Evol. Comput.* 27 (2016) 1–30. URL: <https://doi.org/10.1016/j.swevo.2016.01.004>. doi:10.1016/j.swevo.2016.01.004.
- [3] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, P. N. Suganthan, Ensemble of differential evolution variants, *Inf. Sci.* 423 (2018) 172–186. URL: <https://doi.org/10.1016/j.ins.2017.09.053>. doi:10.1016/j.ins.2017.09.053.
- [4] W. Qian, J. Chai, Z. Xu, Z. Zhang, Differential evolution algorithm with multiple mutation strategies based on roulette wheel selection, *Appl. Intell.* 48 (2018) 3612–3629. URL: <https://doi.org/10.1007/s10489-018-1153-y>. doi:10.1007/s10489-018-1153-y.
- [5] Y. Zheng, X. Xu, H. Ling, S. Chen, A hybrid fireworks optimization method with differential evolution operators, *Neurocomputing* 148 (2015) 75–82. URL: <https://doi.org/10.1016/j.neucom.2012.08.075>. doi:10.1016/j.neucom.2012.08.075.
- [6] C. Dong, W. W. Y. Ng, X. Wang, P. P. K. Chan, D. S. Yeung, An improved differential evolution and its application to determining feature weights in similarity-based clustering, *Neurocomputing* 146 (2014) 95–103. URL: <https://doi.org/10.1016/j.neucom.2014.04.065>. doi:10.1016/j.neucom.2014.04.065.
- [7] R. Mallipeddi, P. N. Suganthan, Q. Pan, M. F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, *Appl. Soft Comput.* 11 (2011) 1679–1696. URL: <https://doi.org/10.1016/j.asoc.2010.04.024>. doi:10.1016/j.asoc.2010.04.024.

- [8] X. Li, L. Wang, Q. Jiang, N. Li, Differential evolution algorithm with multi-population cooperation and multi-strategy integration, *Neurocomputing* 421 (2021) 285–302. URL: <https://doi.org/10.1016/j.neucom.2020.09.007>. doi:10.1016/j.neucom.2020.09.007.
- [9] L. Gui, X. Xia, F. Yu, H. Wu, R. Wu, B. Wei, Y. Zhang, X. Li, G. He, A multi-role based differential evolution, *Swarm Evol. Comput.* 50 (2019). URL: <https://doi.org/10.1016/j.swevo.2019.03.003>. doi:10.1016/j.swevo.2019.03.003.
- [10] S. Kitayama, M. Arakawa, K. Yamazaki, Differential evolution as the global optimization technique and its application to structural optimization, *Appl. Soft Comput.* 11 (2011) 3792–3803. URL: <https://doi.org/10.1016/j.asoc.2011.02.012>. doi:10.1016/j.asoc.2011.02.012.
- [11] A. K. Qin, P. N. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, in: *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005, 2-4 September 2005, Edinburgh, UK, IEEE, 2005*, pp. 1785–1791. URL: <https://doi.org/10.1109/CEC.2005.1554904>. doi:10.1109/CEC.2005.1554904.
- [12] J. Zhang, A. C. Sanderson, JADE: adaptive differential evolution with optional external archive, *IEEE Trans. Evol. Comput.* 13 (2009) 945–958. URL: <https://doi.org/10.1109/TEVC.2009.2014613>. doi:10.1109/TEVC.2009.2014613.
- [13] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Trans. Evol. Comput.* 10 (2006) 646–657. URL: <https://doi.org/10.1109/TEVC.2006.872133>. doi:10.1109/TEVC.2006.872133.
- [14] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, *IEEE Trans. Evol. Comput.* 15 (2011) 55–66. URL: <https://doi.org/10.1109/TEVC.2010.2087271>. doi:10.1109/TEVC.2010.2087271.
- [15] A. W. Iorio, X. Li, Solving rotated multi-objective optimization problems using differential evolution, in: *Australasian joint conference on artificial intelligence, Springer, 2004*, pp. 861–872.
- [16] R. Storn, On the usage of differential evolution for function optimization, in: *Proceedings of North American Fuzzy Information Processing, IEEE, 1996*, pp. 519–523.
- [17] A. K. Qin, V. L. Huang, P. N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, *IEEE transactions on Evolutionary Computation* 13 (2008) 398–417.
- [18] R. Gämperle, S. D. Müller, P. Koumoutsakos, A parameter study for differential evolution, *Advances in intelligent systems, fuzzy systems, evolutionary computation* 10 (2002) 293–298.
- [19] R. Storn, K. V. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, *J. Glob. Optim.* 11 (1997) 341–359. URL: <https://doi.org/10.1023/A:1008202821328>. doi:10.1023/A:1008202821328.
- [20] R. Storn, K. Price, Differential evolution a simple evolution strategy for fast optimization, *Dr. Dobb's Journal* 22 (1997) 18–24.
- [21] J. Ronkkonen, S. Kukkonen, K. V. Price, Real-parameter optimization with differential evolution, in: *2005 IEEE congress on evolutionary computation, volume 1, IEEE, 2005*, pp. 506–513.
- [22] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters

- in differential evolution: A comparative study on numerical benchmark problems, *IEEE transactions on evolutionary computation* 10 (2006) 646–657.
- [23] M. Z. Ali, N. H. Awad, P. N. Suganthan, R. G. Reynolds, An adaptive multipopulation differential evolution with dynamic population reduction, *IEEE transactions on cybernetics* 47 (2016) 2768–2779.
- [24] J. Liu, J. Lampinen, A fuzzy adaptive differential evolution algorithm, *Soft Computing* 9 (2005) 448–462.
- [25] D. Zaharie, D. Petcu, Adaptive pareto differential evolution and its parallelization, in: *International Conference on Parallel Processing and Applied Mathematics*, Springer, 2003, pp. 261–268.
- [26] M. G. Omran, A. Salman, A. P. Engelbrecht, Self-adaptive differential evolution, in: *International conference on computational and information science*, Springer, 2005, pp. 192–199.
- [27] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE transactions on evolutionary computation* 10 (2006) 646–657.
- [28] E. Mezura-Montes, J. Velázquez-Reyes, C. 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*, 2006, pp. 485–492.
- [29] S. M. Islam, S. Das, S. Ghosh, S. Roy, P. N. Suganthan, An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42 (2011) 482–500.
- [30] W. Yi, L. Gao, X. Li, Y. Zhou, A new differential evolution algorithm with a hybrid mutation operator and self-adapting control parameters for global optimization problems, *Applied Intelligence* 42 (2015) 642–660.
- [31] G. Jia, Y. Wang, Z. Cai, Y. Jin, An improved  $(\mu + \lambda)$ -constrained differential evolution for constrained optimization, *Information Sciences* 222 (2013) 302–322.
- [32] N. Hansen, A. Auger, D. Brockhoff, D. Tušar, T. Tušar, Coco: performance assessment, *arXiv preprint arXiv:1605.03560* (2016).