MPADE: An Improved Adaptive Multi-Population Differential Evolution Algorithm Based on JADE

Javier Ramos, Miguel Leon, Ning Xiong
School of Innovation, Design and Engineering
Mälardalen University
Västerås, Sweden
Email: javioverflow@gmail.com,
miguel.leonortiz@mdh.se

***Abstract*—JADE is an state-of-the-art adaptive differential evolution algorithm which implements "DE/current-to-$p$best" as its mutation strategy, adapts its mutation factor and crossover rate, and uses an optional external archive to keep track of potential removed individuals in previous generations. This paper proposes MPADE, which extends JADE by using a multi-populated approach to solve high dimensional real-parameter optimization problems. This mechanism helps preventing the two well-known problems affecting the differential evolution algorithm performance, which are premature convergence and stagnation. The algorithm was tested using the benchmark functions in IEEE Congress on Evolutionary Computation 2014 test suite. MPADE was compared using Wilcoxon test to JADE algorithm and with other state-of-the-art algorithms that either use a multi-population approach or adapt their parameters. The experimental results show that the proposed new algorithm improves significantly its precursor and it is also suggested that other state-of-the-art algorithms could benefit from the multi-populated based approach.**

***Index Terms*—Differential evolution, multi-population, coarse-grained parallel, population topology, adaptive parameter, global optimization, evolutionary optimization.**

I. Introduction

SEARCHING the global optimum over a continuous space is a common task to perform in many real-world applications in technology and science [1], [2], [3]. Price and Storn proposed a new Evolutionary Algorithm (EA) called Differential evolution (DE) [4]. DE is a genetic population based algorithm, which employs a stochastic search method for finding a global opminum over multi-dimensional real valued functions. Although simple and straightforward to implement, it is really powerful and outperforms many other state-of-the-art stochastic algorithms in terms of accuracy and convergence speed [5], even in multi-modal problems [6].

Since the proposal of DE, many great efforts have been made in the scientific community in order to improve its search accuracy, convergence speed [7], [8], parallel capability [9].

DE's performance relies on two main components. First, the mutation and crossover operators employed to generate the new offspring in every generation, whose effectiveness is heavily linked with the type of problem to solve [10]. Next, the three control parameters used in DE: the magnifying factor F of the difference vector during the mutation step, the crossover control parameter CR during the crossover step and the population size NP. These parameters greatly affect DE's performance and they must be tuned depending on the problem to tackle [11], [12].

Two important issues that have to be considered when tuning these parameters are premature convergence and stagnation [13]. Premature convergence happens when the population has converged to local optima whereas the individuals within it are not diverse enough to enable the movement towards the global optimum, e.g. if the mutation operator takes the best individual as reference to generate offspring [10]. Stagnation, on the other hand, occurs when the population cannot generate better individuals even when the population has not converged [14], e.g. an F too high allows jumps too big over the dimension space and lowers the quality of the offspring.

Even if some works suggest possible optimal values for the control parameters [10], [15], different problems types or even different stages during the search may need different parameters settings. New mechanisms have been proposed in order to update them dynamically during the search. They can be classified into the following three categories [16]:

- Deterministic: The logic to update the control parameters is agnostic to the search performance [17].
- Adaptative: The parameters are updated using data from the search [18], [19], [20]. The algorithm proposed in this paper falls in this category.
- Self-adaptative: Instead of having global control parameters, every individual in the population has its own control parameters [21], [22].

Another way to handle the two issues affecting the DE algorithm, premature convergence and stagnation, is adopting a multi-population design, which may offer the opportunity to prevent such problems [23], [24].

Although little research has been made in parallelizing differential evolution algorithm compared to other evolutionary algorithms, promising results have been obtained. Starkweather, Whitley, and Mathias [25] showed that isolated subpopulations usually converge to different solutions and generate better results, especially for separable functions. Also in [25], they observed that multi-populated designs can often get better overall solutions by combining partial solutions through the migration process, mainly in non-separable functions.

Besides its advantages in obtaining better search results, this approach also benefits from the upsides of parallelized and distributed applications [23]. In particular, the approach explained in this paper is considered as coarsed-grained parallelization which is one of the three levels in which a DE can be parallelized [26].

The aim of this work is to show that an arbitrary differential evolution variant can be easily enhanced by adapting it to the multi-populated approach. In this paper, we propose a multi-population approach for a DE variant called JADE[20], which already uses parameter adaptation to improve the search depending on the problem that is being considered. Our enhancement consists on dividing JADE's population into several subpopulations and adapting JADE operations to work in a multi-population setting. This helps tackling premature convergence by diversifying the population in several subpopulations [24] and, at the same time, avoiding stagnation by considering the best individuals of the different subpopulations during the mutation phase.

The rest of the paper is organized as follows. Section II reviews the basic implementation of differential evolution. Related work regarding multi-population and adaptation of control parameters is described in Section III. Then, MPADE is proposed and explained in Section IV. In Section V, experimental results and comparisons with relevant state-of-the-art DE algorithms are presented. Finally, the last section contains concluding remarks and future work.

II. Differential Evolution

This section briefly describes the basic operations of differential evolution and present basic notations and terminology that will be used later on.

Similar to other evolutionary algorithms for numerical optimization, a DE population is represented as a set of real parameter vectors $x_i = (x_1, ..., x_D)$, $i = 1, ..., NP$, where D is the dimensionality of the target problem, and NP is the population size. At the beginning of the search, the individual vectors in the population are randomly initialized according to a uniform distribution $x_j^{MIN} \leq x_{j,i,0} \leq x_j^{MAX}$, where x_j^{MIN} and x_j^{MAX} are the lower and upper boundaries in the search space of the dimension j.

After the initialization phase, DE enters in a loop of evolutionary operations, which are mutation, crossover and selection. It stops when some termination criterion is encountered, e.g. after doing a certain amount of evaluations of the fitness function.

In each generation G, a mutant vector $v_{i,G}$ is generated based on existing population members $x_{i,G}, i \in 1...NP$ by applying some mutation strategy. Examples of these mutation strategies are:

- rand/1

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

- best/1

$$v_{i,G} = x_{best,G} + F * (x_{r1,G} - x_{r2,G}) \tag{2}$$

- current-to-best/1

$$v_{i,G} = x_{i,G} + F * (x_{best,G} - x_{i,G}) + F * (x_{r1,G} - x_{r2,G}) \tag{3}$$

- rand-to-best/1

$$v_{i,G} = x_{r1,G} + F * (x_{best,G} - x_{r1,G}) + F * (x_{r2,G} - x_{r3,G}) \tag{4}$$

The indexes $r1, ..., r5$ are randomly selected integers from the interval $[1, NP]$ such that $i \neq r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5$. $x_{best,G}$ is the best individual in population during generation G. The parameter F falls in the real interval [0,2] and controls the magnitude of the differential mutation operator. Some other mutation strategies that have been proposed recently can be found in [27] and [28].

After generating the mutant vector $v_{i,G}$, it is crossed with the parent $x_{i,G}$ in order to generate trial vector $u_{i,G}$. In the basic DE, the crossover operator used is the binomial crossover. It is implemented as follows:

$$u_{j,i,G} = \begin{cases} v_{j,i,G} & \text{if } rand[0,1) \leq CR \text{ or } j = j_{rand} \\ x_{j,i,G} & \text{otherwise} \end{cases} \tag{5}$$

$rand[0, 1)$ denotes a uniformly selected random real number from $[0, 1)$. j_{rand} is a uniformly randomly selected integer value from $[1, D]$, which ensures that each trial vector $u_{i,G}$ differs at least in the j_{rand} dimension from its counterpart $x_{i,G}$ in the population. The control parameter crossover rate CR, which falls within the real interval [0, 1], determines how many dimension values are copied from the mutant vector.

After all the trial vectors $u_{i,G}, 1 \leq i \leq NP$, have been generated, a selection process determines the survivors. The selection operator in conventional DE compares the fitness of each individual $x_{i,G}$ against its corresponding trial vector $u_{i,G}$, keeping the one with least fitness value in the population for the next generation.

$$x_{i,G+1} = \begin{cases} u_{i,G} & \text{if } f(u_{i,G}) \leq f(x_{i,G}) \\ x_{i,G} & \text{otherwise} \end{cases} \tag{6}$$

being f the function that evaluates the fitness of an individual. Accurately, the fitness measures the error from the proposed solution by the individual to the optimal solution of the function. Therefore, the lower fitness the better is the solution found by the individual.

III. Related Work

In this section, we introduce the main features of JADE, a DE variant which is the basis of our MPADE algorithm. Then, we highlight the multi-populated design and we review some variants of the multi-populated DE variants that have been proposed recently.

A. Review of JADE

JADE [20] is a well-known and effective variant of DE, which updates dynamically the value of the control parameters F and CR during the search. It introduced a new mutation strategy named DE/current-to-$p$best/1, which was adopted later on by many other state-of-the-art DE variants [29], [30]. Optionally, it uses an external archive to increase the diversity of the population.

1) Strategy DE/current-to-pbest/1: This strategy is a variant of the current-to-best/1 strategy where it's convergence capability is adjustable by tuning the parameter p. It has the following expression:

$$v_{i,G} = x_{i,G} + F_i * (x_{pbest,G} - x_{i,G}) + F_i * (x_{r1,G} - x_{r2,G}) \quad (7)$$

F_i determines the mutation factor used by the individual x_i. The parameter p is a random real number inside the interval [0,1] and $x_{pbest,G}$ is a random individual chosen from the top $\lceil N * p \rceil$ best individuals in the generation G.

2) Parameter adaptation: JADE employs the parameter adaptation by assigning to the individual x_i its own CR_i and F_i control parameters. At the beginning of each generation, their values are generated using the expressions presented below, where $randn$ and $randc$ are random generators that use normal and Cauchy distribution, respectively.

$$CR_i = randn_i(\mu_{CR}, 0.1) \quad (8)$$

$$F_i = randc_i(\mu_F, 0.1) \quad (9)$$

and then truncated so they satisfy the following conditions:

$$0 \leq CR_i \leq 1 \quad (10)$$

$$0 < F_i \leq 1 \quad (11)$$

except the case $F_i \leq 0$, where F_i is regenerated.

Regarding μ_{CR} and μ_F, they are initialized with value 0.5 at the beginning of the search and updated in the end of each generation based on the following equations:

$$\mu_{CR} = (1 - c) * \mu_{CR} + c * mean_A(S_{CR}) \quad (12)$$

$$\mu_F = (1 - c) * \mu_F + c * mean_L(S_F) \quad (13)$$

Here, the parameter c is a positive constant in the interval [0,1]. S_{CR} and S_F are the set of all crossover rates CR_i and mutation factors F_i of the trial vectors that survive to the next generation. $mean_A$ is the arithmetic mean and $mean_L$ is the Lehmer mean, whose expression is:

$$mean_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \quad (14)$$

3) External archive: If an external archive is used, a set A with maximum $|NP|$ amount of latest removed individuals is maintained. This archive is used during the mutation phase. When choosing the individual $x_{r2,g}$, the set of individuals to consider is $P \cup A$ instead of P.

B. Multi-populated DE

The main difference of this design compared to the conventional differential evolution resides in how are the individuals organized. Instead of having one big population, multi-populated DE uses many small subpopulations, conventionally named demes [31] or islands [32], that stay isolated during the search except for when the migration phase happens.

Migration is the process where exchanges of information happens among the demes. Migrations can be fired periodically, e.g. every certain amount of generations, which in this case is called migration frequency (MF). The migration phase in the algorithm proposed in this paper is fired following this mechanism. Another way to trigger the migration process is by checking when certain conditions are met, e.g. some of the demes have converged into a local minimum. If two demes exchange information between themselves during the migration phase, they are said to be neighbors.

The topology of a population establish the neighborhood relationships among the demes. Some common topologies are [31]: ring topology, where each deme has two unique neighbors; broadcast, one deme that sends information to the rest; fully connected, where each deme is connect with the rest.

Below we present some multi-populated differential evolution variants of interest that have been proposed in the last years or shown to perform well.

1) PDE: In [9], a new variant (PDE) is proposed. It uses a ring topology of 16 demes. Its migration strategy consists in replacing a random individual different from the best one in a deme by the best individual in the previous deme. At each iteration, every deme generates uniformly a random number in the interval [0,1]. If it is greater than the value of the migration constant ϕ, that deme will perform the migration.

2) MPDE: In [33], another multi-population differential evolution (MPDE) is proposed. In this case, this variant is based in the DE which employs the mutation strategy DE/best/1. In MPDE, the vector $best$ is picked from the subpopulation that the individual which is currently being mutated is in. The migration is also employed during this phase, where the difference vector is composed on individuals of any subpopulation in order to ensure exchange of information. Experimental results showed that this variant performs better than its non-multipopulated counterpart.

3) RRBMPDE: Ishani Chatterjee and Mengchu ZhouIn [34] proposed a new multi-populated differential evolution algorithm. The population is subdivided in two subpopulations which compete to get the best individual by using as mutation strategies DE/rand/1 and DE/rand-to-best/1. When both populations meet the termination condition, the individuals are compared and the best one is picked as the result of the search.

4) SCoPTDE: SCoPTDE [35] is proposed by Yu Sun, a new multi-population differential evolution variant based on the classic mutation strategy DE/rand/1. In this variant, the subpopulations are connected obeying the ring topology [31]. The migrations phase in SCoPTDE consists in replacing the worst individual in the next subpopulation according to the ring topology by the best in the current subpopulation. This operation is employed by every subpopulation in every generation.

IV. MPADE

In this section, we propose a new DE algorithm, MPADE, which is an extension of JADE by using a multi-population approach. Along the following subsections, the differences compared to its predecessor are explained and the new methods are introduced, such as the new migration strategy employed in MPADE .

Algorithm 1: MPADE

```
// Initialization phase
G = 0
Initialize $P_0$ = $(x_{1,0}, ..., x_{NP,0})$ randomly
Set $\mu_F = 0.5$; $\mu_{CR} = 0.5$; $A = \emptyset$
// Main loop
while The termination criteria are not met do
    // Pre-Iteration
    $S_F = \emptyset$; $S_{CR} = \emptyset$
    for $i = 1$ to $NP$ do
        $CR_i = randn(\mu_{CR}, 0.2)$; $F_i = randn(\mu_F, 0.05)$
        // Mutation phase
        Randomly choose $x_{pBest,G}$
        Randomly choose $y_{r1,G}$ and $[y_{r2,G}||a_{r_3,G}]$
        $v_{i,G} = x_{pBest,G} + F_i * (y_{r1,G} - [y_{r2,G}||a_{r_3,G}])$
        // Crossover phase
        Generate $j_{rand} = randint(1, D)$
        for $j = 1$ to $D$ do
            if $j = j_{rand}$ or $rand(0,1) < CR_i$ then
                $u_{j,i,G} = v_{j,i,G}$
            else
                $u_{j,i,G} = x_{j,i,G}$
            end
        end
        // Selection phase
        if $f(x_{i,G}) \leq f(u_{i,G})$ then
            $x_{i,G+1} = x_{i,G}$
        else
            $x_{i,G+1} = u_{i,G}$
            $x_{i,G} \to A$; $CR_i \to S_{CR}$; $F_i \to S_F$
        end
    end
    $G = G + 1$
    // Migration phase
    if G mod MF = 0 then
        for $i = 1$ to NP do
            for $j = 1$ to $D$ do
                Choose random subpopulations Y and Z
                $y_{best,j,G}$ = best individual from Y
                $z_{best,j,G}$ = best individual from Z
                $x_{i,j,G} = 0.5 * y_{best,j,G} + 0.5 * z_{best,j,G}$
            end
        end
    end
    // Post-Iteration
    Randomly truncate A so $|A| \leq NA$
    $\mu_{CR} = (1 - c) * \mu_{CR} + c * mean_A(S_{CR})$
    $\mu_F = (1 - c) * \mu_F + c * mean_A(S_F)$
end
```

A. Parameter Adaptation

In MPADE , the control parameters F and CR are adapted during the search. The adaptation is performed globally in the whole population, similar as it was performed in JADE:

- At the beginning of each iteration, F_i and CR_i are generated using a random generator based on the normal distribution (F_i is no longer generated by the Cauchy distribution):

$$F_i = randn_i(\mu_F, 0.05) \tag{15}$$

$$CR_i = randn_i(\mu_{CR}, 0.2) \tag{16}$$

- At the end of each generation, μ_F and μ_{CR} are updated using the arithmetic mean, as described below. Note that the Lehmer mean was replaced by the arithmetic mean when μ_F is updated.

$$\mu_F = (1 - c) * \mu_F + c * mean_A(S_F) \tag{17}$$

$$\mu_{CR} = (1 - c) * \mu_{CR} + c * mean_A(S_{CR}) \tag{18}$$

B. Archive

An archive A of size NA is created and maintained exactly as in JADE. When an individual has to be replaced due to being less fit than the trial vector, instead of removing it, it's added to the archive set. In the end of every generation, if the amount of individuals in the archive is above NA, the archive is truncated i.e. individuals from the archive are randomly removed so as to keep $|A| \leq NA$. The archive is used during the mutation phase in order to get more diversity in the new generations. This mechanism is described in the Subsection IV-C.

C. Mutation Strategy

The mutation strategy used by MPADE is based on DE/Current-to-pbest, which is the one employed in JADE.

During the mutation phase, a trial individual $v_{i,G}$ is generated for every individual $x_{i,G}$ in each subpopulation using the following strategy:

$$v_{i,G} = x_{pBest,G} + F * (y_{r1,G} - \alpha) \tag{19}$$

$$\alpha = \begin{cases} y_{r2,G} & \text{if } r_\alpha \leq 0.5 \\ a_{r3,G} & \text{otherwise} \end{cases} \tag{20}$$

The $pbest$ vector individual $x_{pBest,G}$ is created exactly in the same way as in JADE, where the whole population is considered and one individual is chosen randomly among the p best individuals.

Denoting Y the subpopulation which $x_{i,G}$ belongs to, the indexes $r1$ and $r2$ are random integers in the interval $[1, NSP_Y]$, where NSP_Y is the number of individuals in the subpopulation Y. The index $r3$ is a random integer with $r3 \in [1, NA]$ and r_α is a random floating value between 0 and 1.

D. Migration Strategy

In MPADE , the migration strategy consists of exchange of information in various dimension levels among individuals belonging to different subpopulations.

After every MF generations, the following migration strategy is performed for every dimension in every individual:

$$x_{i,j,G} = \frac{y_{best,j,G} + z_{best,j,G}}{2} \tag{21}$$

where y_{best} and z_{best} are the best individuals in the subpopulations Y and Z, respectively. These subpopulations are chosen randomly and are unique, which means, $Y \neq Z$.

V. Experimental Results

MPADE was evaluated on the CEC2014 benchmark problem set [36] and compared with the following state-of-the-art DE algorithms: JADE [20], SaDE [19], PDE [9], MPDE [33], RRBMPDE [34] and SCoPTDE [35].

A. Experimental Settings

Regarding the DE variants selected for the comparison, we used the control parameters which were shown to perform the best in the experiments conducted in the cited papers.

Regarding MPADE, we used the following values:

- Initial F = 0.5 and CR = 0.5.
- The population size $NP = 100$, composed of 15 subpopulations with 5 of them having 6 individuals and 10 of them having 7 individuals.
- Migration frequency parameter $MF = 100$.
- Archive size $NA = 100$.

We evaluated MPADE and the other DE variants on the 30 benchmark problems from *CEC2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization* benchmark suite [36].

The CEC'14 test suite is composed of the following kinds of functions:

- $f1 \sim f3$ are unimodal.
- $f4 \sim f16$ are multimodal.
- $f17 \sim f22$ are hybrid. These are functions whose variables are divided into subcomponents, each one of these using a basic function.
- $f23 \sim f30$ are composition. This type of functions merge the properties of sub-functions maintaining continuity around the optima.

The guidelines of the CEC2014 benchmark competition were followed [36]. Since the functions in the experiments had 30 dimensions, the maximum number of fitness function calls per execution was $D * 10,000 = 300,000$. On every dimension, the lower and upper bounds in the search space were -100 and 100, respectively. Furthermore, a difference of fitness value below 10^{-8} with respect to the optimal solution was considered to be 0.

B. Comparison with other adaptive DE variants

The results shown in Table I were obtained by calculating the mean of the results obtained from executions on each function in the CEC 2014 benchmark test suite 30 times for every algorithm. The symbols $+$, $-$ and $\approx$ are used to indicate that a given DE variant performed better ($+$), worse ($-$), or equivalently ($\approx$) compared to MPADE.

TABLE I: Comparison of MPADE with JADE and SaDE, state-of-the-art adaptive Differential Evolution variants. The CEC2014 test suite was used as benchmark.

F	MPADE	JADE	SaDE
f_1	**6.05E+04**	5.46E+05 (-)	6.07E+05 (-)
f_2	**0.00E+00**	**0.00E+00** ($\approx$)	**0.00E+00** ($\approx$)
f_3	**0.00E+00**	2.25E-02 (-)	**0.00E+00** ($\approx$)
f_4	**2.38E-01**	2.25E+00 (-)	5.24E+00 (-)
f_5	**2.02E+01**	2.04E+01 (-)	2.09E+01 (-)
f_6	**1.53E-01**	9.89E+00 (-)	2.70E+00 (-)
f_7	**2.47E-04**	0.00E+00 (-)	5.25E-03 (-)
f_8	8.22E+00	**0.00E+00** (+)	4.97E-01 (+)
f_9	**1.46E+01**	3.50E+01 (-)	3.18E+01 (-)
f_{10}	5.66E+01	**3.93E+00** (+)	1.17E+02 (-)
f_{11}	**7.80E+02**	2.28E+03 (-)	5.60E+03 (-)
f_{12}	**4.94E-01**	5.59E-01 (-)	1.86E+00 (-)
f_{13}	**1.23E-01**	2.42E-01 (-)	2.62E-01 (-)
f_{14}	3.40E-01	**2.53E-01** (+)	2.55E-01 (-)
f_{15}	**2.93E+00**	3.76E+00 (-)	8.93E+00 (-)
f_{16}	**7.88E+00**	1.03E+01 (-)	1.19E+01 (-)
f_{17}	1.26E+04	1.30E+05 (-)	**3.62E+03** (+)
f_{18}	6.85E+02	1.94E+03 (-)	**9.80E+01** (+)
f_{19}	**2.46E+00**	5.18E+00 (-)	5.15E+00 (-)
f_{20}	**3.04E+01**	3.42E+03 (-)	1.58E+02 (-)
f_{21}	4.22E+03	8.49E+04 (-)	**8.92E+02** (+)
f_{22}	1.29E+02	1.51E+02 (-)	**1.06E+02** (+)
f_{23}	**3.15E+02**	**3.15E+02** ($\approx$)	**3.15E+02** ($\approx$)
f_{24}	**2.23E+02**	2.25E+02 (-)	2.25E+02 (-)
f_{25}	**2.04E+02**	**2.04E+02** ($\approx$)	2.06E+02 (-)
f_{26}	1.03E+02	**1.00E+02** (+)	1.04E+02 (-)
f_{27}	**3.03E+02**	3.33E+02 (-)	3.68E+02 (-)
f_{28}	8.10E+02	**7.78E+02** (+)	8.72E+02 (-)
f_{29}	1.05E+03	**9.13E+02** (+)	2.04E+03 (-)
f_{30}	**1.46E+03**	1.91E+03 (-)	1.56E+03 (-)
	+	6	5
	-	21	22
	$\approx$	3	3

Recall that the fitness measures the error from the proposed solution to the optimal solution of the function. Therefore, the lower fitness the better is the solution found by a particular algorithm.

According to this, at a glance, it can be observed in Table I that MPADE outperforms JADE and SaDE in most of the CEC2014 benchmark functions.

Since MPADE is based on JADE, it's especially interesting to study thoroughly the differences between them. As shown in the Table I, MPADE outperforms JADE in all the unimodal ($f_1 \sim f_3$) and hybrid ($f_{17} \sim f_{22}$) functions, except the function f_2 for which both algorithms found the optimum. The reason of this significant enhancement can be found in the subpopulations. Since hybrid functions are divided into subcomponents, multi-populated differential evolution solutions have advantage since one subpopulation can focus on one subcomponent at a time [25]. Regarding multimodal functions

($f_4 \sim f_{16}$), MPADE also performs particularly well compared to JADE. In 10 out of the 13 functions, MPADE obtained better solutions. On the other three functions, the biggest differences in results are in f_8 and f_{10}, the only separable functions in the multimodal subset, and the rest of them are non-separable. Having this in mind, it's evident that the multi-populated algorithm performs better in those non-separable functions that are multimodal. In the last group containing composite functions $f_{17} \sim f_{22}$, the search performance of MPADE was similar to that of JADE.

SaDE [19] is another well-known state-of-the-art differential evolution variant that employs an adaptive mechanism over the control parameters. JADE cannot significantly outperform SaDE in the CEC2014 benchmark suite. However, it's interesting to notice that after extending JADE using a multi-population approach, it becomes statistically better. SaDE gets outperformed by MPADE in all the unimodal, multimodal and composite functions, except in the separable function f_8, in which MPADE comparatively has inferior performance. In functions f_2, f_3 and f_{23} both algorithms got the same results. Concerning the hybrid functions subset $f_{17} \sim f_{22}$, SaDE was outperforming JADE in every function. Although SaDE still outperforms MPADE in overall, now it gets weaker than MPADE in 2 of the functions.

JADE [20] and SaDE [19] were also compared to MPADE using the Wilcoxon signed ranks test. The results show that MPADE is significantly better than JADE as well as SaDE when using a significance level of $\alpha = 0.05$. The exact p-values are presented in Table II.

TABLE II: Wilcoxon test on CEC2014 benchmarks of MPADE over JADE and SaDE adaptive DE variants

MPADE	JADE	SaDE
R^+	332.5	317.5
R^-	102.5	117.5
p-value	0.012508	0.02933

C. Comparison with other multi-population DE variants

MPADE search performance was also compared with other state-of-the-art multi-population differential evolution variants. These are PDE [9], MPDE [33], RRBMPDE [34] and SCoPTDE [35]. As we can see from the results of this comparison in Table IV, MPADE obtained the best results in 19 of the 30 benchmark functions. Identically as in Table I, the symbols $+$, $-$ and $\approx$ are used to indicate that a given DE variant performed better ($+$), worse ($-$), or equivalently ($\approx$) compared to MPADE.

Besides this overall analysis, it is useful to compare MPADE with the multi-population variants according to a specific type of problems to understand its upsides, especially with the new hybrid functions that were introduced in the CEC benchmark set of 2014. Regarding the unimodal functions $f_1 \sim f_3$, MPADE performs similarly to RRBMPDE, except in the function f_1 in which MPADE is better. In the multimodal functions $f_4 \sim f_{16}$, MPADE outperforms the rest of the algorithms in 9 out of 13. Concerning the hybrid functions $f_{17} \sim f_{22}$, our algorithm performs relatively well getting the best solutions in half of them. Nevertheless, RRBMPDE was slightly better than MPADE in the composite functions $f_{23} \sim f_{30}$.

We carried out an overall comparison using Wilcoxon signed ranks test and it was shown that MPADE significantly outperformed the other DE variants by using a significance level of $\alpha = 0.05$. The exact p-values can be found in Table III.

TABLE III: Wilcoxon test on CEC2014 benchmarks of MPADE over PDE [9], MPDE [33], RRBMPDE [34] and SCoPTDE [35] multi-population DE variants

MPADE	PDE	MPDE	RRBMPDE	SCoPTDE
R^+	434.5	464.0	361.0	330.5
R^-	30.5	1.0	104.0	104.5
p-value	0.000031	0.000002	0.00781	0.013859

D. Study in fitness evolution along generations

In an effort to gain a deeper understanding about what are the consequences of applying the migration strategy in MPADE, data about the variations in the fitness of the individuals in every subpopulation were gathered.

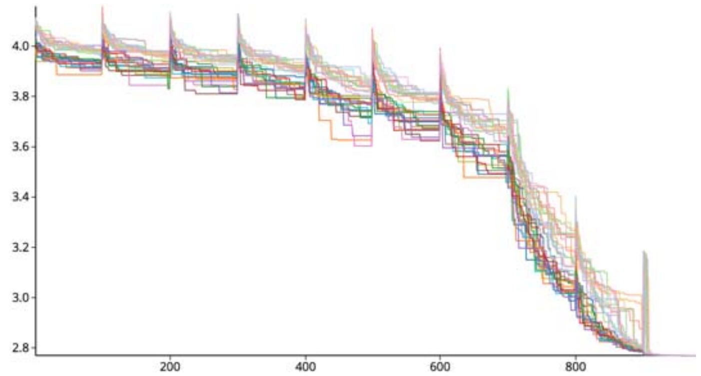
The fitness of the best individual in every subpopulation is normalized using a log10 scale (Y axis) and is plotted for every generation (X axis). The consequent charts can be examined in Figures 1 and 2. By taking Fig. 1, one can easily observe peaks in the fitness of all the subpopulations every 100 generations approximately. This is not a trivial feature of MPADE. This is the effect of employing the migration strategy with migration rate of 100. During every migration phase, a massive exchange of information is carried out in a dimension level among the individuals in all the subpopulations. This process produces huge increments in the population diversity, a Genetics phenomenon that translates into these fitness noises.

In addition to its ability to keep diversity, it also contributes to finding better solutions. Function f_{11} gives a clear example of this. Note that MPADE has been shown to have the best search performance in this specific function compared to the other algorithms that are used in this paper. As can be observed in Figure 1, just after the migration in the 700th generation, a huge improvement in the current solutions can be achieved.

In order to conduct a more detailed study of these phenomena, the fitness of the worst and the best individuals from an arbitrary subpopulation was tracked during the execution on function f_{19}. In Figure 2 we can observe the variations from the 1500th generation to the 1650th. At the end of every migration phase, i.e. the following generations after the 1500th and 1600th, the difference between worst and the best fitness gets bigger. The benefits from this mechanism are that better and worse solutions are discovered, and while the algorithm keeps the better ones for next generations thanks to the selection phase, the worst individuals are quickly fixed after few generations, as it can be seen in the plot.

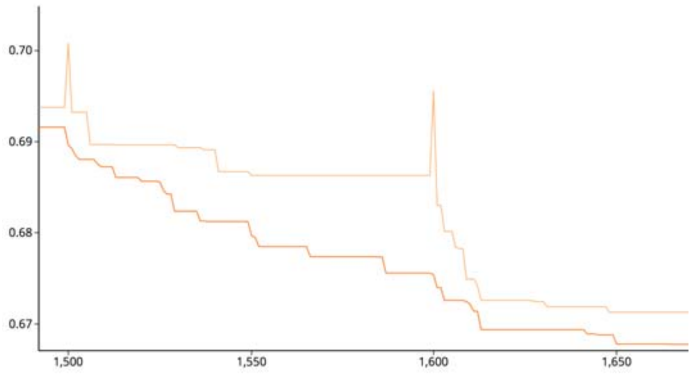
TABLE IV: Comparison of MPADE with PDE [9], MPDE [33], RRBMPDE [34] and SCoPTDE [35], state-of-the-art multi-population Differential Evolution variants. The CEC2014 test suite was used as benchmark.

F	MPADE	PDE	MPDE	RRBMPDE	SCoPTDE
f_1	**6.05E+04**	1.66E+06 (-)	2.30E+08 (-)	3.02E+05 (-)	4.46E+05 (-)
f_2	**0.00E+00**	1.98E-07 (-)	2.46E+10 (-)	**0.00E+00** (≈)	3.74E+03 (-)
f_3	**0.00E+00**	3.85E-05 (-)	6.80E+04 (-)	**0.00E+00** (≈)	1.83E+02 (-)
f_4	**2.38E-01**	6.26E+01 (-)	2.68E+03 (-)	6.27E+01 (-)	6.09E+01 (-)
f_5	2.02E+01	2.05E+01 (-)	**2.00E+01** (+)	2.09E+01 (-)	2.05E+01 (-)
f_6	**1.53E-01**	4.88E+00 (-)	3.30E+01 (-)	1.93E-01 (-)	7.14E+00 (-)
f_7	2.47E-04	7.40E-04 (-)	2.33E+02 (-)	**0.00E+00** (+)	1.78E-02 (-)
f_8	**8.22E+00**	1.03E+01 (-)	1.26E+02 (-)	1.26E+01 (-)	1.64E+01 (-)
f_9	**1.46E+01**	4.51E+01 (-)	1.68E+02 (-)	1.37E+02 (-)	4.44E+01 (-)
f_{10}	**5.66E+01**	1.38E+02 (-)	2.56E+03 (-)	1.65E+02 (-)	2.34E+02 (-)
f_{11}	**7.80E+02**	2.77E+03 (-)	3.41E+03 (-)	6.35E+03 (-)	2.87E+03 (-)
f_{12}	4.94E-01	**3.07E-01** (+)	7.29E-01 (-)	2.18E+00 (-)	4.02E-01 (+)
f_{13}	**1.23E-01**	2.72E-01 (-)	3.78E+00 (-)	3.03E-01 (-)	2.28E-01 (-)
f_{14}	3.40E-01	2.79E-01 (+)	7.52E+01 (-)	2.61E-01 (+)	**2.16E-01** (+)
f_{15}	**2.93E+00**	5.28E+00 (-)	6.16E+04 (-)	1.35E+01 (-)	4.66E+00 (-)
f_{16}	**7.88E+00**	1.07E+01 (-)	1.18E+01 (-)	1.16E+01 (-)	1.01E+01 (-)
f_{17}	**1.26E+04**	1.41E+05 (-)	8.08E+06 (-)	4.48E+04 (-)	1.47E+04 (-)
f_{18}	6.85E+02	2.28E+03 (-)	2.44E+08 (-)	3.23E+02 (+)	**1.37E+02** (+)
f_{19}	**2.46E+00**	4.70E+00 (-)	1.60E+02 (-)	5.52E+00 (-)	4.58E+00 (-)
f_{20}	**3.04E+01**	6.33E+01 (-)	3.08E+04 (-)	6.67E+01 (-)	1.27E+02 (-)
f_{21}	4.22E+03	2.55E+04 (-)	2.81E+06 (-)	5.55E+03 (-)	**1.92E+03** (+)
f_{22}	1.29E+02	2.15E+02 (-)	6.99E+02 (-)	**1.04E+02** (+)	3.83E+02 (-)
f_{23}	**3.15E+02**	**3.15E+02** (≈)	4.63E+02 (-)	**3.15E+02** (≈)	**3.15E+02** (≈)
f_{24}	**2.23E+02**	2.24E+02 (-)	3.08E+02 (-)	**2.23E+02** (≈)	2.26E+02 (-)
f_{25}	**2.04E+02**	**2.04E+02** (≈)	2.28E+02 (-)	2.05E+02 (-)	2.06E+02 (-)
f_{26}	1.03E+02	**1.00E+02** (+)	1.11E+02 (-)	**1.00E+02** (+)	**1.00E+02** (+)
f_{27}	3.03E+02	4.00E+02 (-)	7.93E+02 (-)	**3.00E+02** (+)	4.13E+02 (-)
f_{28}	8.10E+02	8.76E+02 (-)	2.51E+03 (-)	**7.95E+02** (+)	8.91E+02 (-)
f_{29}	**1.05E+03**	1.41E+03 (-)	2.92E+07 (-)	1.22E+03 (-)	9.66E+02 (-)
f_{30}	1.46E+03	2.05E+03 (-)	2.64E+05 (-)	2.10E+03 (-)	**1.34E+03** (+)
	+	3	1	7	6
	-	25	29	19	23
	≈	2	0	4	1



(a) f_{11}

Fig. 1: Zoom in fitness evolution for function f_{11} from generation 0 to 1000. Best and worst individuals of each subpopulation are plotted with high and low saturated colors, respectively.



(a) f_{19}

Fig. 2: Zoom in fitness evolution for the second subpopulation in the function f_{19} from generation 1450 to 1700. Best and worst individual are plotted with a high and low saturated color, respectively.

VI. Conclusion

The purpose of this work is to show how the performance of a particular DE variant, in this case JADE, can be enhanced by redesigning it using a multi-population design. This modification helps avoidinf premature convergence and stagnation in many scenarios [23], [24].

MPADE is proposed in this paper, which is a multi-populated parallel adaptive DE algorithm that extends JADE [20] by using a multi-population approach.

We studied the search performance of MPADE on the 30 benchmark functions from the CEC2014 Special Session on Real-Parameter Single Objective Optimization benchmark suite test [36]. Furthermore, it was compared with the other state-of-the-art DE variants that implement adaptive parameter control, multi-population parallel design, or both. These include SaDE [19], PDE [9], MPDE [33], RRBMPDE [34] and SCoPTDE [35].

The experimental results showed that MPADE outperforms its counterpart, JADE, as well as the other adaptive and multi-populated metholds. Our algorithm performs especially well on the hybrid functions, where the variables are divided into subcomponents. These facts may lead to the thought that each subpopulation works at the same time in only one subcomponent of the problem and the partial solutions are merged during the migration phase [25].

Future works will aim to extend other DE variants by employing a multi-population approach and investigate its performance enhancements. Meanwhile, the design of generic guidelines able to enhance any DE variant by taking advantage of the multi-populated approach will be carried out.

References

[1] K. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media, 2006.

[2] J. Zhang, V. Avasarala, and R. Subbu, "Evolutionary optimization of transition probability matrices for credit decision-making," *European Journal of Operational Research*, vol. 200, no. 2, pp. 557–567, 2010.

[3] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2011.

[4] R. Storn and K. 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.

[5] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.

[6] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 666–685, 2013.

[7] M. Leon and N. Xiong, "A new differential evolution algorithm with alopex-based local search," in *International Conference in Artificial Intelligence and Soft Computing (ICAISC)*, 2016, pp. 420–431.

[8] N. Noman and H. Iba, "Accelerating differential evolution using an adaptative local search," in *IEEE Transactions on Evolutionary Computation*, vol. 12, 2008, pp. 107 – 125.

[9] D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Parallel differential evolution," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 2. IEEE, 2004, pp. 2023–2029.

[10] E. Mezura-Montes, J. Velázquez-Reyes, and 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*. ACM, 2006, pp. 485–492.

[11] J. Liu, "On setting the control parameter of the differential evolution method," in *Proceedings of the 8th international conference on soft computing (MENDEL 2002)*, 2002, pp. 11–18.

[12] R. Mallipeddi and P. N. Suganthan, "Empirical study on the effect of population size on differential evolution algorithm," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 3663–3670.

[13] N. Padhye, P. Mittal, and K. Deb, "Differential evolution: Performances and analyses," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 1960–1967.

[14] J. Lampinen, I. Zelinka, *et al.*, "On stagnation of the differential evolution algorithm," in *Proceedings of MENDEL*, 2000, pp. 76–83.

[15] R. Gämperle, S. D. Müller, and P. Koumoutsakos, "A parameter study for differential evolution," *Advances in intelligent systems, fuzzy systems, evolutionary computation*, vol. 10, pp. 293–298, 2002.

[16] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on evolutionary computation*, vol. 3, no. 2, pp. 124–141, 1999.

[17] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[18] M. Leon and N. Xiong, "Adapting differential evolution algorithms for continuous optimization via greedy adjustment of control parameters," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 6, no. 2, pp. 103–118, 2016.

[19] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

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

[21] H. A. Abbass, "The self-adaptive pareto differential evolution algorithm," in *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, vol. 1. IEEE, 2002, pp. 831–836.

[22] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, vol. 10, no. 8, pp. 673–686, 2006.

[23] D. Zaharie and D. Petcu, "Parallel implementation of multi-population differential evolution," *Concurrent information processing and computing*, vol. 195, p. 223, 2005.

[24] H. Braun, "On solving travelling salesman problems by genetic algorithms," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1990, pp. 129–133.

[25] T. Starkweather, D. Whitley, and K. Mathias, "Optimization using distributed genetic algorithms," in *International Conference on Parallel Problem Solving from Nature*. Springer, 1990, pp. 176–185.

[26] K. Miettinen and P. Preface By-Neittaanmaki, *Evolutionary algorithms in engineering and computer science: recent advances in genetic algorithms, evolution strategies, evolutionary programming, GE*. John Wiley & Sons, Inc., 1999.

[27] W. Qian, J. Chai, Z. Xu, and Z. Zhang, "Differential evolution algorithm with multiple mutation strategies based on roulette wheel selection," *Applied Intelligence*, pp. 1–18, 2018.

[28] M. Leon and N. Xiong, "Alopex-based mutation strategy in differential evolution," in *In Evolutionary Computation (CEC), 2017 IEEE Congress on*, 2017, pp. 1978–1984.

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

[30] 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.

[31] E. Cantú-Paz, "A survey of parallel genetic algorithms," *Calculateurs paralleles, reseaux et systems repartis*, vol. 10, no. 2, pp. 141–171, 1998.

[32] M. Ruciński, D. Izzo, and F. Biscani, "On the impact of the migration topology on the island model," *Parallel Computing*, vol. 36, no. 10, pp. 555–571, 2010.

[33] W.-j. Yu and J. Zhang, "Multi-population differential evolution with adaptive parameter control for global optimization," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 2011, pp. 1093–1098.

[34] I. Chatterjee and M. Zhou, "Differential evolution algorithms under multi-population strategy," in *Wireless and Optical Communication Conference (WOCC), 2017 26th*. IEEE, 2017, pp. 1–7.

[35] Y. Sun, "Symbiosis co-evolutionary population topology differential evolution," in *Computational Intelligence and Security (CIS), 2016 12th International Conference on*. IEEE, 2016, pp. 530–533.

[36] J. Liang, B. Qu, and P. Suganthan, "Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization," *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, 2013.