LSHADE with Semi-Parameter Adaptation Hybrid with CMA-ES for Solving CEC 2017 Benchmark Problems

Ali W. Mohamed
Operations Research Department, Institute of Statistical Studies and Research, Cairo University
Giza 12613, Egypt
aliwagdy@gmail.com

Anas A. Hadi, Anas M. Fattouh, Kamal M. Jambi
College of Computing and Information Technology, King Abdulaziz University, P. O. Box 80200, Jeddah 21589, Saudi Arabia
anas1401@gmail.com

Abstract—**To improve the optimization performance of LSHADE algorithm, an alternative adaptation approach for the selection of control parameters is proposed. The proposed algorithm, named LSHADE-SPA, uses a new semi-parameter adaptation approach to effectively adapt the values of the scaling factor of the Differential evolution algorithm. The proposed approach consists of two different settings for two control parameters F and Cr. The benefit of this approach is to prove that the semi-adaptive algorithm is better than pure random algorithm or fully adaptive or self-adaptive algorithm. To enhance the performance of our algorithm, we also introduced a hybridization framework named LSHADE-SPACMA between LSHADE-SPA and a modified version of CMA-ES. The modified version of CMA-ES undergoes the crossover operation to improve the exploration capability of the proposed framework. In LSHADE-SPACMA both algorithms will work simultaneously on the same population, but more populations will be assigned gradually to the better performance algorithm. In order to verify and analyze the performance of both LSHADE-SPA and LSHADE-SPACMA, Numerical experiments on a set of 30 test problems from the CEC2017 benchmark for 10, 30, 50 and 100 dimensions, including a comparison with LSHADE algorithm are executed. Experimental results indicate that in terms of robustness, stability, and quality of the solution obtained, of both LSHADE-SPA and LSHADE-SPACMA are better than LSHADE algorithm, especially as the dimension increases.**

Keywords— Numerical Optimization; Differential Evolution; LSHADE; Parameter adaptation

I. Introduction

Differential Evolution (DE), proposed by Storn and Price [1], is a stochastic population-based search method. It exhibits excellent capability in solving a wide range of optimization problems with different characteristics from several fields and many real-world application problems [2]. Similar to all other Evolutionary algorithms (EAs), the evolutionary process of DE uses mutations, crossover and selection operators at each generation to reach the global optimum [3, 4]. The performance of DE basically depends on the mutation strategy, the crossover operator. Besides, The intrinsic control parameters (population size *NP*, scaling factor *F*, the crossover rate *Cr*) play a vital role in balancing the diversity of population and convergence speed of the algorithm. The advantages are simplicity of implementation, reliable, speed and robustness [4]. Thus, it has been widely applied in solving many real-world applications of science and engineering [5]. However, DE has many weaknesses, as all other evolutionary search techniques do w.r.t the NFL theorem [6]. Generally, DE has a good global exploration ability that can reach the region of global optimum, but it is slow at exploitation of the solution [7]. Additionally, the parameters of DE are problem-dependent and it is difficult to adjust them for different problems. Moreover, DE performance decreases as search space dimensionality increases [8]. Finally, the performance of DE deteriorates significantly when the problems of premature convergence and/or stagnation occur [8, 9]. Consequently, researchers have suggested many techniques to improve the basic DE. From the literature [8, 10], these proposed modifications, improvements and developments on DE focus on adjusting control parameters. These approaches can be divided into two main groups: The first group focuses on pure random selection of parameter values from random distributions such as the uniform distribution, normal distribution, and Cauchy distribution [11]. Alternatively, the parameters values are changing with the progress of generations using increasing / decreasing linear or nonlinear deterministic function such as [12, 14]. The second group focuses on adjusting control parameters in an adaptive or self-adaptive manner. For instance, Zhang and Sanderson [15] introduced a new differential evolution (DE) algorithm, named JADE, to improve optimization performance by implementing a new mutation strategy ‘‘*DE/current-to-pbest*’’ with optional external archive and by updating control parameters in an adaptive manner. Simulation results show that JADE was better than, or at least competitive to, other classic or adaptive DE algorithms such as Particle swarm and other evolutionary algorithms from the literature in terms of convergence performance. Tanabe and Fukunaga [16] proposed an improved variant of the JADE algorithm [14] and called the same as the Success History based DE (SHADE). In SHADE, instead of sampling the *F* and *Cr* values from gradually adapted probability distributions, the authors used historical memory

978-1-5090-4601-0/17/$31.00 ©2017 IEEE

archives M_{Cr} and M_F which store a set of *Cr* and *F* values, respectively that have performed well in the recent past .The algorithm generates new *Cr*, *F* pairs by directly sampling the parameter space close to one of the stored pairs. Out of the 21 algorithms that participated in the IEEE CEC competition on real parameter single-objective optimization [17], SHADE ranked 3rd, the first two ranks being taken by non-DE based algorithms. Tanabe and Fukunaga [18] further improved the SHADE algorithm by using the linear population size reduction and called this variants LSHADE. In LSHADE, the population size of DE is continually reduced by means of a linear function [6]. In this paper, in order to significantly enhance the performance of LSHADE algorithm, a new semi-parameter adaptation approach is proposed. This approach consists of two different settings for two control parameters *F* and *Cr*. The first control parameter settings are the LSHADE adaptation, which is used to adapt *Cr* only while *F* is randomly generated from uniform distribution for the first half of function evaluations. The second control parameter setting is the LSHADE adaptation, which is used to adapt *F* and *Cr* for the second half of function evaluations. In fact, after conducting many experiments to observe LSHADE adaptation process, taking into consideration that the adaptation process of *Cr* is almost stopped and no further changing in *Cr* values in the second half of function evaluations. Thus, the core idea of the proposed semi-adaptation approach for the crossover rate *CR* and *F* is based on the following empirical principle. The semi-adaptive algorithm is better than pure random algorithm or fully adaptive or self-adaptive algorithm. The pure random algorithm cannot solve efficiently or effectively the optimization problems within limited number of function evaluations due to the fact that it enhances population diversity and global exploration capability of the algorithm while the exploitation tendency may be significantly deteriorated as there is no bias in any specific search direction i.e. it is unable to concentrates the exploitation of some sub-regions of the search space. On the other hand, the adaptive or self-adaptive algorithm can balance the exploration ability and exploitation tendency of the algorithm with no guarantee of escaping local solutions or avoiding stagnation due to systematic adaptation which may be keep the same values for many generations without changes. In general, *F* is an important parameter that controls the evolving rate of the population i.e. it is closely related to the convergence speed [19]. A small *F* value encourages the exploitation tendency of the algorithm that makes the search focus on neighborhood of the current solutions; hence it can enhance the convergence speed. However, it may also lead to premature convergence [20]. On the other hand, A large *F* value improves the exploration capability of the algorithm that can make the mutant vectors distribute widely in the search space and can increase the diversity of the population [21]. However, it may slow down the search [20]. The constant crossover *Cr* reflects the probability, with which the trial individual inherits the actual individual's genes i.e. which and how many components are mutated in each element of the current population [22], [23]. The constant crossover *Cr* practically controls the diversity of the population [20]. Recently, the authors in [24] claim that a key parameter affecting the operation of DE is the *Cr*. Additionally, *Cr* is usually more sensitive to problems with different characteristics such as unimodality and multimodality, separable and non-separable problems. Consequently, it is logically deduced that in the first half of the search process, the LSHADE adaptation, is used to adapt *Cr* to considerably control the diversity of the algorithm in solving many types of optimization problems and to preserve the stability of the algorithm. However, in order to enrich the search space and balance both exploration capability and exploitation tendency of the algorithm, *F* is randomly generated from uniform distribution. Then, during the second half of the search process, the LSHADE adaptation is used to adapt *F* to significantly control and concentrate the convergence behavior of the algorithm on a sub-region of the search space while the population diversity reaches an appropriate level using LSHADE adaptation for *Cr*.

On the other hand, a hybridized EA algorithm is likely to be more powerful than single EA algorithm in solving many hard optimization problems. Thus, in order to enhance the performance capability of DE, many local and global optimizers have been hybridized with DE algorithm [6]. In this paper, a modified version of CMA-ES is introduced. The modified version of CMA-ES undergoes the crossover operation to improve the exploration capability of the proposed framework.

The remainder of the paper is organized as follows. The proposed LSHADE-SPACMA is presented in Section II. The experimental set-up and simulation results are presented in section III. Finally, section IV summarizes the conclusions of this work.

II. LSHADE WITH SEMI-PARAMETER ADAPTATION HYBRID WITH CMA-ES (LSHADE-SPACMA)

In this section, we will describe the details of LSHADE with semi-parameter adaptation hybrid with CMA-ES (LSHADE-SPACMA) which is a new improvement of LSHADE. We will describe LSHADE and CMA-ES algorithms. After that, we will discuss the hybridization framework and the semi-parameter adaptation process.

A. LSHADE

In order to establish a starting point for the optimization process, an initial population P^0 must be created. Typically, each j^{th} component $(j = 1,2,\ldots\ldots,D)$ of the i^{th} individuals $(i = 1,2,\ldots\ldots,NP)$ in the P^0 is obtained as follow:

$$x_{j,i}^{0} = x_{j,L} + \mathrm{rand}(0,1)\left(x_{j,U} - x_{j,L}\right) \tag{1}$$

Where rand (0,1) returns a uniformly distributed random number in [0, 1].

At generation G, for each target vector x_i^G, a mutant vector v_i^G is generated according to current-to-pbest/1 mutation strategy which was proposed by in the framework of JADE by Zhang and A. C. Sanderson [15].

$$v_i^G = x_i^G + F_i^G\left(x_{pbest}^G - x_i^G\right) + F(x_{r1}^G - x_{r2}^G) \tag{2}$$

The *P* value here is considered as a control parameter for the greediness of the mutation strategy in order to balance exploitation and exploration. r_1 is a random index selected from the population. r_2 is another random index selected from the concatenation of the population with an external archive. This external archive holds parent vectors which successfully produced better vectors. x_{pbest}^{G} is the best individual vector with the best fitness value in the population at generation *G*. The scale factor F_i^G is a positive control parameter for scaling the difference vector.

In the crossover, the target vector is mixed with the mutated vector, using the following scheme, to yield the trial vector u_i^G.

$$u_{i,j}^{G} = \begin{cases} v_{i,j}^{G} \ , if \ (rand_{i,j} \leq Cr \ OR \ j = \ j_{rand}) \\ \quad x_{i,j}^{G} \ , \ otherwise \end{cases} \tag{3}$$

Where $rand_{j,i}\ i \in \{1, N\}\ and\ j\ \in \{1\ , D\}$ is a uniformly distributed random number in [0,1], $Cr \in [0,1]$ called the crossover rate that controls how many components are inherited from the mutant vector, j_{rand} is a uniformly distributed random integer in[1, *D*] that makes sure at least one component of trial vector is inherited from the mutant vector.

DE adapts a greedy selection strategy. If and only if the trial vector u_i^G yields as good as or a better fitness function value than x_i^G, then u_i^G is set to x_i^{G+1}. Otherwise, the old vector $x_i^G$is reserved. The selection scheme is as follows (for a minimization problem):

$$x_i^{G+1} = \begin{cases} u_i^G \ \ , if \ (f(u_i^G) \leq \ f(x_i^G)) \\ \quad x_i^G \ , \ otherwise \end{cases} \tag{4}$$

In order to improve the performance of LSHADE-SPA, Linear Population Size Reduction (LPSR) was used. In LPSR the population size will be decreased according to a linear function. The linear function in LSHADE-SPA was:

$$N_{G+1} = round[\left(\frac{N^{min}\_N^{init}}{MAX_{NFE}}\right) * NFE + \ N^{init}] \tag{5}$$

Where *NFE* is the current number of fitness evaluations, *MAX_NFE* is the maximum number of fitness evaluations, Ninit is the initial population size, and *Nmin* = 4 which is the minimum number of individuals that DE can work with.

B. CMA-ES

Among many variants of Evolution Strategies, CMA-ES was efficiently able to solve diverse types of optimization problems [25]. In CMA-ES the search space is modeled using multivariate normal distribution. New individuals are generated using Gaussian distribution considering the path that the population takes over generations. CMA-ES automatically adapt the mean vector m, covariance matrix C, and step size σ.

CMA-ES steps are as the following:

1. Create an initial population and evaluate the fitness function.
2. Generate new individuals using Gaussian distribution:

$$x_i = N(m, \sigma^2 C) \ \forall \ i = 1{:}n \tag{6}$$

3. Mean vector m is updated using best μ individuals according to: $m = \sum_{i=1}^{\mu} w_i\, x_i$ where $\sum_{i=1}^{\mu} w_i \ = 1$ and $w1 \ \geq \ w2 \ \geq \ldots \geq \ w\mu$.
4. Step size σ and Covariance matrix C are updated.
5. Repeat steps 2 to 3 until a stopping criterion is met.

For more details about CMA-ES are illustrated in [25].CMA-ES MATLAB code used in this paper was downloaded from [26].

C. Semi-Parameter adaptation of Scaling Factor (F) and Crossover Rate (Cr)

Parameter setting has a significant impact on the performance of DE. The practices in the fields of parameter adaptation demonstrate the relationship between the problem itself and the parameter values [5]. Each problem has its own appropriate parameter values. In order to perform Semi-Parameter Adaptation (SPA) for F and *Cr*, SPA is composed of two parts. The first part will be activated during the first half of the search, while the second part will be activated during the second half of the search.

a) First part of SPA

The idea is to activate the change one parameter at a time policy. Thus, during the first part of SPA, the adaptation will be concentrated on one parameter *Cr* using original LSHADE adaptation, while *F* parameter will be generated using uniform distribution randomly within a specific limit. The first part of SPA using the condition: (*nfes* < *max_nfes/2*) where *nfes* is the current number of function evaluations and *max_nfes* is the maximum number of function evaluation.

During SPA, each individual has its own F_i and Cr_i values. F_i will be generated using uniform distribution within the range (0.45,0.55):

$$F_i = \ 0.45 + 0.1 * rand \tag{7}$$

On the other hand, Cr_i values is adapted according to the following equation:

$$Cr_i = \ randn(Mcr_i, 0.1) \tag{8}$$

Where Mcr_i is a randomly selected memory slot where successful means of previous generations which are stored. Memory index i is selected randomly from the range [1,*h*] where h here is the memory size. Initially, all Mcr values are set to 0.5, and by the end of each generation, one memory slot Mcr is updated using the arithmetic mean of Cr_i values, which successfully generate new individuals.

b) Second part of SPA

During the second part, L-SHADE adaptation will be used to adapt *F* parameter adaptation using the following equation. The adaptation will be concentrated on *F*. Each individual has its own F_i value. F_i will be generated using Cauchy distribution:

$$F_i = \ randc(MF_i, \sigma) \tag{9}$$

σ is the standard deviation for Cauchy distribution and it was set to 0.1, MF_i is a randomly selected memory slot where successful means of previous generations which are stored. By the end of each generation, one memory slot MF is updated using the Lehmer mean of F_i values, which successfully generate new individuals. F_i values of the last 5 generation of the first part of SPA are used to initialize memory slot MF_i for the second part of SPA.

Cr parameter adaptation process will remain as is during the second this part. Nonetheless, due to the nature of LSHADE parameter adaptation, *Cr* parameter will be gradually frozen to the adapted values. According to LSHADE parameter adaptation, when all *Cr* values in a generation are failed to generate successful individuals, the corresponding memory slot is set to a terminal value. Thus, *Mcr* will be frozen until the end of the search.

D. LSHADE-SPACMA Hybridization Framework

In order to improve the performance of LSHADE-SPA, we implement a hybridization framework between LSHADE-SPA and a modified version of CMA-ES. Here we add a crossover operation to CMA-ES in order to improve the exploration capability of the proposed framework. Crossover equation Eq.3 were applied after the CMA-ES offspring generation step.

Fig.1 shows LSHADE-SPACMA pseudo code. The framework starts with a mutual population *P*. Each individual *x* in *P* will generate offspring individual u using either LSHADE or CMA-ES. This assignment is done according to class probability variable (*FCP*). *FCP* values are randomly selected from memory slots M_{FCP}. By the end of each generation, one memory slot M_{FCP} is updated according to the performance of each algorithm. Thus, more populations will be assigned gradually to the better performance algorithm. The update is performed using individuals that successfully generate new individuals only. Memory slot M_{FCP} is updated according to:

$$M_{FCP,g+1} = (1-c)M_{FCP,g} + c\ \Delta_{Alg1} \tag{10}$$

where c is the learning rate, and Δ_{Alg1} is the improvement rate for each algorithm.

$$\Delta_{Alg1} = \min(0.8, \max(0.2, \frac{\omega_{Alg1}}{\omega_{Alg1} + \omega_{Alg2}})) \tag{11}$$

Where 0.2 and 0.8 values are the minimum and maximum probabilities assigned to each algorithm. Thus, to maintain both algorithms executed simultaneously, FCP values will be always kept in the range (0.2, 0.8). ω_{Alg1} is the summation of differences between old and new fitness values for each individual belongs algorithm $Alg1$.

$$\omega_{Alg1} = \sum_{i=1}^{n} f(x) - f(u) \tag{12}$$

Where f is the fitness function, x is the old individual, u is the offspring individual, and n is the number of individuals belongs to algorithm $Alg1$.

III. EXPERIMENTAL RESULTS

A. Numerical benchmarks

The performance of the proposed LSHADE-SPA algorithm is evaluated using a set of problems presented in the CEC2017 competition on real-parameter single objective optimization. This benchmark contains 30 test functions with a diverse set of characteristics. *D* is the dimensionality of the problem and the functions are tested on 10*D*, 30*D*, 50*D* and 100*D*. In summary, functions 1-3 are unimodal, functions 4-10 are multimodal, functions 11-20 are hybrid functions and 21-30 are composition functions. More details can be found in [27].

Algorithm: LSHADE_SPACMA algorithm

```
G = 1, N_G = N^init, Archive A = ∅; nfes=0;
Initialize population P_G = (x_1,G, ..., x_N,G) randomly;
Set all values in M_CR, M_F, M_FCP to 0.5;
Initialize CMA parameters;
while termination criteria is not met do
     S_CR = ∅, S_F = ∅, S_FCP = ∅;
     for i = 1 to N do
          r_i = Select from [1,H] randomly;
          CR_i,G = randn_i(M_CR,ri , 0.1);
          FCP_i,G = M_FCP,ri ;
          If nfes <  max_nfes/2
               F_i,G = 0.45+0.1*rand;
          else
               F_i,G = randc_i(M_F,ri , 0.1);
          end
     end
     [P_LSHADE,G ,P_CMA,G]=Split(P_G ,FCP_G);
     V_G,LSHADE  = Generate donor vectors using LSHADE ;
     V_G,CMA     = Generate donor vectors using CMA;
     V_G = Concatenate( V_G,LSHADE , V_G,CMA);
     U_G = Generate trial vectors( V_G , CR_G);
     Evaluate U_G;
     Update nfes;
     Update P_G according to the evaluation of U_G
     Store successful FCP_G , F_G, and CR_G
     Update archive A
     If (archive size> |A|)
          delete randomly selected individuals archive
     end
     Update memory M_CR, M_FCP;
     If nfes >  max_nfes/2
          Update memory M_F
     end
     Calculate N_G+1 according to Eq. (5);
     if N_G < N_G+1 then
          Sort individuals in P based on their fitness
          values and delete lowest N_G − N_G+1 members;
          Resize archive size |A| according to new |P|;
     end
     Update CMA parameters;
     G++
end
```

Fig. 1. Pseudo-code of LSHADE-SPACMA algorithm

B. Algorithm parameters

Algorithm parameters for LSHADE-SPACMA are as the following. The initial population size (*NP*) were set to 18 * *D*, Pbest individuals rate (*Pbest*), Memory size *(H),* and archive rate (*Arc_rate*) were set to (0.11, 1.4, 5) as it was described in LSHADE. For the hybridization, Probability Variable (F_{CP}) was set to 0.5, and learning rate (c) was set to 0.8. The threshold, where the second part of SPA is activated, was set to (*max_nfes/2*).

C. Algorithm complexity

All experiments were implemented and executed using MATLAB R2014a running on a PC with core i7-4790 (3.60 GHz) CPU and 12 GB RAM running using win 8.1 OS. In order to evaluate the computational complexity of LSHADE-SPACMA as described in [27], the time needed (T0) to run the following test problem where calculated:

```
for i=1:1000000
    x=x + x; x=x/2; x=x*x;
    x=sqrt(x); x=log(x);
    x=exp(x); x=x/(x+2);
end
```

Table I shows the computed algorithm complexity on 10, 30 and 50 dimensions. *T1* is the time to execute 200,000 evaluations of benchmark function *f18* by itself with *D* dimensions, and *T2* is the time to execute LSHADE-SAP with 200,000 evaluations of *f18* in *D* dimensions.

TABLE I. ALGORITHM COMPLEXITY RESULTS

	T0	T1	T2	(T2 −T1) /T0
D = 10	**0.1093**	**0.8391**	**2.1835**	**12.30009**
D = 20		**1.0570**	**3.6724**	**23.92864**
D = 30		**1.4338**	**3.7066**	**20.79414**
D = 50		**3.0237**	**7.7564**	**43.30009**

D. Statistical results

To evaluate the performance of algorithms, experiments were conducted on the test suite. We adopt the solution error measure $f(x) - f(x^*)$, where x is the best solution obtained by algorithms in one run and x^* is the well-known global optimum of each benchmark function. Error values and standard deviations smaller than 10-8 are taken as zero [26]. The maximum number of function evaluations (FEs), the terminal criteria, is set to $10000 \times D$, all experiments for each function and each algorithm run 51 times independently. The statistical results of the LSHADE-SPACMA on the benchmarks with 10, 30, 50 and 100 dimensions are summarized in Tables II-V. It includes the obtained best, worst, median, mean values and the standard deviations of error from the optimum solution of the proposed LSHADE-SPACMA over 51 runs for all 30 benchmark functions.

E. Comparison against LSHADE algorithm

The statistical results of the comparisons on the benchmarks with 10, 30, 50 and 100 dimensions are summarized in Table VI. It includes the obtained best and the standard deviations of error from the optimum solution of LSHADE-SPACMA, LSHADE-SPA, and LSHADE, as taken a baseline algorithm, over 51 runs for all 30 benchmark functions. The same control parameter values that were suggested in the original paper [18] were used to run the test. The best results are marked in bold for all problems. It can be observed from Table VII in the appendix that In 10 dimensions, all algorithms almost show comparable performance. However, it can be obviously observed that the superiority of the LSHADE-SPACMA and LSHADE-SPA algorithm against LSHADE algorithm considerably increases as the dimension of the problems increases from 10 to 100 dimensions. Furthermore, it can be clearly deduced that LSHADE-SPACMA is competitive with and in most cases superior to LSHADE-SPA, especially as dimensions increases.

Thus, both LSHADE-SPACMA and LSHADE-SPA performed equal to or in most cases better than the winner of the 2014 competition (LSHADE). Therefore, from the above results and comparisons, it is clearly visible that the proposed semi-parameter adaptation approach considerably balances the global exploration ability and local exploitation tendency for the majority of multimodal, hybrid and composition functions on all dimensions much more than the LSHADE adaptation approach for crossover rate and scaling factor. On the other hand, it is noteworthy to mentioning that the performance of LSHADE-SPA is significantly improved because it is adaptively combined with the proposed modified CMA-ES with crossover operation which yields LSHADE-SPACMA framework.

To compare and analyze the solution quality from a statistical angle of different algorithms and to check the behavior of the stochastic algorithms [28], the results are compared using non-parametric statistical hypothesis tests: multi-problem Wilcoxon signed-rank test (to check the differences between two algorithms for all functions); at a 0.05 significance level. All the p values in this paper were computed using SPSS 20.00.

Table VI summarizes the statistical analysis results of applying Wilcoxon's test between LSHADE-SPACMA, LSHADE-SPA and LSHADE algorithms on the functions with different dimensions. Where R^+ is the sum of ranks for the functions in which the first algorithm outperforms the second algorithm in the row, and R^- is the sum of ranks for the opposite. Larger ranks indicate larger performance discrepancy. For each of the competitive functions in this table, the numbers in Better, Equal and worse columns denote the number of problems in which the first algorithm is better than, equal or worse than the second algorithm. Based on the result of the test, one of three signs (+, - , and ≈) is assigned for the comparison of any two algorithms (shown in the last column), where (+) sign means the first algorithm is significantly better than the second, (-) sign means the first algorithm is significantly worse than the second, and (≈) sign means that there is no significant difference between the two algorithms. The p values under the significance level are shown in bold. From the results shown in Table VI, we can see that LSHADE-SPACMA provides higher R+ values than R- in all the dimensions. Moreover, LSHADE-SPACMA is significantly better than LSHADE and LSHAD-SPA in D=30 ,50 and 100 and D=100, respectively, which indicate that LSHADE-SPACMA clearly has the best overall performance and significantly improves upon both the performance of LSHADE-SPA and LSHADE algorithms as the number of dimensions increases. Besides, LSHADE-SPA is significantly better than LSHADE in D=30, 50 and 100 which also proves that LSHADE with the semi-adaptive approach is much better than LSHADE with the fully adaptive approach. Moreover, the proposed modified version of CMA-ES with crossover approach is significantly enhanced the LSHADE-SPA performance.

TABLE II. RESULTS OF THE 10D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F5	2.62E-06	2.99E+00	1.00E+00	1.35E+00	7.13E-01
F6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F7	1.04E+01	1.17E+01	1.09E+01	1.10E+01	3.51E-01
F8	0.00E+00	2.98E+00	9.95E-01	1.04E+00	7.44E-01
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	1.95E-01	1.25E+02	3.75E+00	1.05E+01	2.40E+01
F11	0.00E+00	9.95E-01	0.00E+00	1.95E-02	1.39E-01
F12	0.00E+00	3.37E+02	1.20E+02	1.14E+02	6.57E+01
F13	0.00E+00	5.95E+00	4.84E+00	3.37E+00	2.42E+00
F14	0.00E+00	9.95E-01	0.00E+00	5.31E-02	1.95E-01
F15	3.19E-03	1.49E+00	4.85E-01	4.11E-01	2.66E-01
F16	1.96E-01	1.12E+00	6.99E-01	6.42E-01	2.30E-01
F17	1.97E-02	2.10E+01	3.32E-01	1.54E+00	4.44E+00
F18	1.16E-02	2.05E+01	4.68E-01	1.97E+00	5.44E+00
F19	0.00E+00	1.09E+00	2.00E-02	8.13E-02	2.08E-01
F20	0.00E+00	6.24E-01	3.12E-01	1.84E-01	1.67E-01
F21	1.00E+02	2.05E+02	1.38E+02	1.54E+02	4.93E+01
F22	1.00E+02	1.00E+02	1.00E+02	1.00E+02	7.82E-02
F23	3.00E+02	3.04E+02	3.03E+02	3.02E+02	1.50E+00
F24	1.00E+02	3.33E+02	3.30E+02	2.92E+02	8.29E+01
F25	3.98E+02	4.46E+02	4.43E+02	4.26E+02	2.24E+01
F26	3.00E+02	3.00E+02	3.00E+02	3.00E+02	0.00E+00
F27	3.73E+02	3.95E+02	3.88E+02	3.88E+02	3.13E+00
F28	3.00E+02	4.73E+02	3.00E+02	3.63E+02	8.29E+01
F29	2.27E+02	2.36E+02	2.30E+02	2.30E+02	2.24E+00
F30	2.02E+02	3.37E+02	2.17E+02	2.30E+02	3.25E+01

TABLE III. RESULTS OF THE 30D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F5	0.00E+00	1.01E+01	2.98E+00	3.69E+00	2.48E+00
F6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F7	3.20E+01	3.61E+01	3.37E+01	3.38E+01	8.24E-01
F8	0.00E+00	8.95E+00	3.98E+00	3.59E+00	1.62E+00
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	6.65E+02	2.36E+03	1.41E+03	1.44E+03	3.18E+02
F11	0.00E+00	1.19E+01	2.98E+00	3.79E+00	2.49E+00
F12	1.26E+02	1.47E+03	5.57E+02	5.54E+02	2.53E+02
F13	1.99E+00	2.21E+01	1.62E+01	1.52E+01	4.71E+00
F14	2.00E+01	2.63E+01	2.30E+01	2.28E+01	1.78E+00
F15	2.93E-01	1.33E+01	4.47E+00	5.37E+00	2.80E+00
F16	2.21E+00	2.20E+02	1.44E+01	3.77E+01	5.54E+01
F17	1.44E+01	5.93E+01	3.04E+01	3.17E+01	8.68E+00
F18	2.06E+01	2.87E+01	2.25E+01	2.34E+01	2.00E+00
F19	4.70E+00	1.42E+01	9.08E+00	9.45E+00	2.01E+00
F20	2.31E+01	2.05E+02	5.33E+01	8.16E+01	5.86E+01
F21	2.00E+02	2.16E+02	2.09E+02	2.09E+02	4.19E+00
F22	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.00E+00
F23	3.46E+02	3.65E+02	3.57E+02	3.57E+02	4.07E+00
F24	4.24E+02	4.32E+02	4.29E+02	4.29E+02	2.38E+00
F25	3.78E+02	3.78E+02	3.78E+02	3.78E+02	1.22E-02
F26	8.68E+02	1.03E+03	9.48E+02	9.43E+02	3.56E+01
F27	4.29E+02	5.00E+02	5.00E+02	4.97E+02	1.41E+01
F28	3.00E+02	4.12E+02	3.00E+02	3.34E+02	5.12E+01
F29	3.56E+02	4.81E+02	3.71E+02	3.95E+02	4.06E+01
F30	2.76E+02	6.24E+02	3.97E+02	4.01E+02	9.06E+01

TABLE IV. RESULTS OF THE 50D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F4	0.00E+00	5.07E+00	0.00E+00	5.68E-01	1.45E+00
F5	9.95E-01	1.09E+01	5.97E+00	6.22E+00	2.10E+00
F6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F7	5.51E+01	5.93E+01	5.70E+01	5.71E+01	1.07E+00
F8	1.99E+00	1.09E+01	5.97E+00	5.85E+00	2.29E+00
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	2.41E+03	5.26E+03	3.55E+03	3.61E+03	6.70E+02
F11	3.98E+00	1.69E+01	8.95E+00	9.17E+00	3.37E+00
F12	5.30E+02	2.38E+03	1.52E+03	1.58E+03	4.43E+02
F13	3.98E+00	6.87E+01	4.03E+01	3.53E+01	1.52E+01
F14	2.10E+01	3.79E+01	2.80E+01	2.89E+01	3.92E+00
F15	6.58E+00	3.46E+01	1.40E+01	1.57E+01	6.71E+00
F16	1.27E+02	8.60E+02	3.64E+02	4.09E+02	1.68E+02
F17	5.23E+01	6.62E+02	2.34E+02	2.94E+02	1.19E+02
F18	2.14E+01	5.93E+01	3.23E+01	3.32E+01	7.23E+00
F19	1.50E+01	3.87E+01	2.08E+01	2.18E+01	4.83E+00
F20	6.77E+01	4.64E+02	1.12E+02	1.63E+02	1.08E+02
F21	2.05E+02	2.37E+02	2.09E+02	2.15E+02	9.22E+00
F22	1.00E+02	4.63E+03	1.00E+02	8.17E+02	1.49E+03
F23	4.19E+02	4.52E+02	4.40E+02	4.39E+02	7.25E+00
F24	5.04E+02	5.30E+02	5.11E+02	5.13E+02	5.80E+00
F25	4.31E+02	4.79E+02	4.64E+02	4.63E+02	8.49E+00
F26	8.42E+02	1.24E+03	1.14E+03	1.14E+03	6.99E+01
F27	5.00E+02	5.00E+02	5.00E+02	5.00E+02	8.98E-05
F28	4.39E+02	5.00E+02	4.39E+02	4.40E+02	8.51E+00
F29	2.37E+02	4.86E+02	2.51E+02	2.75E+02	5.95E+01
F30	2.66E+02	1.93E+03	4.74E+02	6.50E+02	4.06E+02

TABLE V. RESULTS OF THE 100D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	0.00E+00	9.28E+01	0.00E+00	3.13E+00	1.59E+01
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F4	7.94E+01	1.40E+02	1.10E+02	1.12E+02	9.24E+00
F5	5.97E+00	1.69E+01	9.95E+00	1.08E+01	2.58E+00
F6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F7	1.08E+02	1.15E+02	1.12E+02	1.12E+02	1.56E+00
F8	5.97E+00	1.79E+01	1.09E+01	1.13E+01	2.94E+00
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	7.66E+03	1.19E+04	9.52E+03	9.53E+03	9.00E+02
F11	1.79E+01	1.01E+02	3.35E+01	3.54E+01	1.31E+01
F12	3.33E+03	8.47E+03	4.82E+03	4.83E+03	8.05E+02
F13	6.67E+01	2.92E+02	1.26E+02	1.30E+02	4.05E+01
F14	5.48E+01	1.12E+02	7.47E+01	7.77E+01	1.38E+01
F15	6.21E+01	2.04E+02	1.01E+02	1.09E+02	3.35E+01
F16	4.21E+02	2.24E+03	1.37E+03	1.35E+03	4.00E+02
F17	4.58E+02	1.97E+03	9.73E+02	9.88E+02	3.36E+02
F18	7.86E+01	2.01E+02	1.27E+02	1.32E+02	2.43E+01
F19	5.18E+01	1.06E+02	7.10E+01	7.41E+01	1.21E+01
F20	4.74E+02	2.17E+03	1.33E+03	1.28E+03	3.59E+02
F21	2.34E+02	3.13E+02	2.42E+02	2.43E+02	1.06E+01
F22	6.41E+03	1.14E+04	9.04E+03	9.00E+03	9.34E+02
F23	5.64E+02	5.98E+02	5.82E+02	5.82E+02	7.68E+00
F24	8.89E+02	9.59E+02	9.11E+02	9.16E+02	1.63E+01
F25	6.11E+02	7.65E+02	6.76E+02	6.72E+02	4.30E+01
F26	3.00E+03	3.29E+03	3.15E+03	3.14E+03	7.12E+01
F27	5.00E+02	5.00E+02	5.00E+02	5.00E+02	1.34E-04
F28	5.00E+02	5.00E+02	5.00E+02	5.00E+02	1.07E-04
F29	6.64E+02	1.94E+03	1.24E+03	1.25E+03	2.69E+02
F30	4.11E+02	7.55E+02	5.54E+02	5.62E+02	6.28E+01

TABLE VI. WILCOXON'S TEST BETWEEN LSHADSPACMA , LSHADE-SPA AND LSHADE ALGORITHM OVERALL DIMENSIONS

D	Algorithms	R^+	R^-	p-value	+	≈	-	Dec.
10	SPACMA vs LSHADE	155.5	97.5	0.346	12	8	10	≈
	SPA vs LSHADE	144	109	0.570	12	8	10	≈
	SPACMA vs SPA	167.5	63.5	0.071	12	9	9	≈
30	SPACMA vs LSHADE	236.5	88.5	0.046	15	9	6	+
	SPA vs LSHADE	171.5	59.5	0.049	15	9	6	+
	SPACMA vs SPA	149	104	0.465	11	8	11	≈
50	SPACMA vs LSHADE	260	65	0.009	20	5	5	+
	SPA vs LSHADE	261.5	89.5	0.029	17	4	9	+
	SPACMA vs SPA	233	118	0.143	18	4	8	≈
100	SPACMA vs LSHADE	427	8	0.000	28	1	1	+
	SPA vs LSHADE	348	58	0.001	21	2	7	+
	SPACMA vs SPA	316	62	0.002	21	3	6	+

IV. CONCLUSION

In order to enhance the overall performance of LSHADE algorithm, introducing new adaptation approaches to adjusting control parameters crossover rate and scaling factor is a must although it is a challenging task. Besides, for achieving superior performance, DE may be intelligently hybridized with appropriate optimizers.Therefore, in this paper, an alternative adaptation approach for the selection of control parameters is proposed. The proposed algorithm, named LSHADE-SPA, uses a new semi-parameter adaptation approach based on randomization and adaptation to effectively adapt the values of the scaling factor of the Differential evolution algorithm during the search process. Besides, LSHADE-SPA is hybridized with a modified version of CMA-ES. The modified version of CMA-ES undergoes the crossover operation to improve the exploration capability of the proposed LSHADE-SPACMA algorithm. The proposed algorithms were tested on the benchmarks of the CEC2017 which is used in the special Session and Competition on Real-Parameter Single Objective Optimization of the IEEE CEC2017. As a summary of results, the performance of the LSHADE-SPACMA algorithm was statistically superior to and competitive with both LSHADE-SPA and LSHADE algorithms in the majority of functions and for different dimensions especially for high dimension functions with different types.

REFERENCES

[1] R. Storn and K. Price, "Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces," International Computer Science Institute Technical Report, Tech. Rep. TR-95-012, 1995..

[2] K . Price, R . Storn, and J. Lampinen, "Differential evolution: a practical approach to global optimization", Heidelberg: Springer; 2005.

[3] X. Li, M. Yin, "Modified differential evolution with self-adaptive parameters method," J Comb Optim, 2014: 31(2): 546-576.

[4] W. Yong, L. Han-Xiong, H. Tingwen, and L. Long, "Differential evolution based on covariance matrix learning and bimodal distribution parameter setting", Applied Soft Computing, Volume 18, May 2014, Pages 232-247.

[5] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution-an updated survey", Swarm and Evolutionary Computation Vol. 27, 2016, Pages 1–30.

[6] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," in IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 67-82, Apr 1997.

[7] N. Noman, and H. Iba, "Accelerating differential evolution using an adaptive local search", IEEE Trans Evol Comput 2008;12(1):107–25.

[8] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator", IEEE Transactions on Evolutionary Computation, 2009, 13(3), 526-553..

[9] F. Neri, , and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis", Artificial Intelligence Review, 2010, 33(1-2), 61-106.

[10] S. Das, and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art". IEEE transactions on evolutionary computation, 15(1), 4-31, 2011.

[11] T. C. Chiang, C. N. Chen, and Y. C. Lin, "Parameter control mechanisms in differential evolution: a tutorial review and taxonomy". In Differential Evolution (SDE), 2013 IEEE Symposium, 2013.

[12] A. Draa, S. Bouzoubia, and I. Boukhalfa, "A sinusoidal differential evolution algorithm for numerical optimization", Appl. Soft Comput. 27 (2015) 99–126.

[13] A. W. Mohamed, "An improved differential evolution algorithm with triangular mutation for global numerical optimization". Computers & Industrial Engineering, 85, 359-375, 2015.

[14] S. Das, A. Konar, and U.K. Chakraborty, "Two improved differential evolution schemes for faster global search," in Proc. GECCO, Jun.2005,pp. 991-998.

[15] J. Zhang, and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," IEEE Transactions on Evolutionary Computation, 13(5): 945-958, 2009.

[16] R. Tanabe, A. Fukunaga, "Success-history based parameter adaptation for differential evolution", In Proceedings of the IEEE Congresson Evolutionary Computation 2013, June 20–23, Cancún, México, 2013, pp.71–78.

[17] J. J. Liang, B. Y. Qin, P .N. Suganthan, A. G. Hernandez-Diaz, "Problem definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization", Zhengzhou University/Nanyang Technological University, Zhengzhou, China/Singapore, 2013.

[18] R. Tanabe, A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction", In Proceedings of IEEE Congress on Evolutionary Computation 2014, July 6–11, Beijing, China, 2014, pp.1658–1665.

[19] 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(2), 398–417, 2009

[20] V. Feoktistov, "Differential Evolution: In Search of Solutions", Berlin, Germany, Springer-verlag, 2006.

[21] Y. Wang, Z. Cai, and Q. Zhang, "Differential Evolution with Composite Trial Vector Generation Strategies and Control Parameters", IEEE Transactions on Evolutionary Computation, 15(1):55–66, 2011.

[22] A. W. Mohamed, and H. Z. Sabry, "Constrained optimization based on modified differential evolution algorithm", Information Sciences, 2012, pp. 171-208.

[23] 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, 2011,11(2), 1679–1696.

[24] J. Montgomery and S. Chen, "An analysis of the operation of differential evolution at high and low crossover rates," IEEE Congress on Evolutionary Computation, Barcelona, 2010, pp. 1-8.

[25] N. Hansen, "The CMA evolution strategy: a comparing review" in Towards a new evolutionary computation: advances on estimation of distribution algorithms, J. A. Lozano, Ed. Springer, 2006, pp. 75–102.

[26] N.,Hansen, and A., Ostermeier, (2009). Cma-es source code [Online]. Available: http://www.lri.fr/~hansen/cmaes_inmatlab.html

[27] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization," Technical Report, Nanyang Technological University, Singapore, November 2016.

[28] S. García, D. Molina, M. Lozano, and F. Herrera,"A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behavior: A case study on the CEC'2005 special session on real parameter optimization", Journal of Heuristics, 2009, Springer, 15, 617–644.

TABLE VII. COMPARISON BETWEEN LSHADE, LSHADE-SPA, AND LSHADE-SPACMA ON THE BENCHMARKS WITH 10, 30, 50 AND 100 DIMENSIONS

	D=10			D=30			D=50			D=100		
	LSHADE	**SPA**	**SPACMA**	**LSHADE**	**SPA**	**SPACMA**	**LSHADE**	**SPA**	**SPACMA**	**LSHADE**	**SPA**	**SPACMA**
F1	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**
F2	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	7.8E+4±2.0E+5	1.9E+2±5.8E+2	**3.1E+0±1.6E+1**
F3	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	2.9E-6±2.9E-6	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**
F4	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	5.9E+1±0.0E+0	5.9E+1±0.0E+0	**0.0E+0±0.0E+0**	6.3E+1±5.0E+1	5.0E+1±4.1E+1	**5.7E-1±1.4E+0**	2.0E+2±3.9E+0	2.0E+2±1.1E+1	**1.1E+2±9.2E+0**
F5	3.0E+0±1.0E+0	1.8E+0±7.9E-1	**1.3E+0±7.1E-1**	6.7E+0±1.2E+0	1.2E+1±2.0E+0	**3.7E+0±2.5E+0**	1.2E+1±2.1E+0	2.9E+1±5.1E+0	**6.2E+0±2.1E+0**	2.8E+1±5.1E+0	5.2E+1±1.1E+1	**1.1E+1±2.6E+0**
F6	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	1.5E-8±4.1E-8	**0.0E+0±2.7E-8**	**0.0E+0±0.0E+0**	7.6E-8±1.5E-7	2.6E-7±5.2E-7	**0.0E+0±0.0E+0**	1.7E-3±9.4E-4	2.6E-5±1.0E-5	**0.0E+0±0.0E+0**
F7	1.2E+1±6.6E-1	1.2E+1±5.8E-1	**1.1E+1±3.5E-1**	3.7E+1±1.1E+0	4.3E+1±2.2E+0	**3.4E+1±8.2E-1**	6.4E+1±2.0E+0	8.0E+1±5.8E+0	**5.7E+1±1.1E+0**	1.4E+2±3.2E+0	1.6E+2±7.0E+0	**1.1E+2±1.6E+0**
F8	2.4E+0±1.1E+0	1.9E+0±7.9E-1	**1.0E+0±7.4E-1**	8.0E+0±1.4E+0	1.3E+1±2.1E+0	**3.6E+0±1.6E+0**	1.2E+1±1.8E+0	2.9E+1±5.4E+0	**5.9E+0±2.3E+0**	2.7E+1±5.1E+0	5.0E+1±1.1E+1	**1.1E+1±2.9E+0**
F9	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**	1.1E-1±1.9E-1	**0.0E+0±0.0E+0**	**0.0E+0±0.0E+0**
F10	2.2E+1±3.7E+1	2.2E+1±3.8E+1	**1.1E+1±2.4E+1**	1.4E+3±2.5E+2	**1.3E+3±2.6E+2**	1.4E+3±3.2E+2	3.1E+3±2.6E+2	**3.0E+3±3.1E+2**	3.6E+3±6.7E+2	1.0E+4±6.4E+2	9.9E+3±4.9E+2	**9.5E+3±9.0E+2**
F11	4.1E-1±7.0E-1	**0.0E+0±0.0E+0**	2.0E-2±1.4E-1	3.4E+1±3.1E+1	1.5E+1±2.3E+1	**3.8E+0±2.5E+0**	5.0E+1±7.7E+0	2.7E+1±1.6E+0	**9.2E+0±3.4E+0**	3.8E+2±1.2E+2	4.4E+1±2.5E+1	**3.5E+1±1.3E+1**
F12	7.7E+1±7.1E+1	1.2E+2±8.0E+1	**1.1E+2±6.6E+1**	1.0E+3±3.4E+2	**4.1E+2±2.4E+2**	5.5E+2±2.5E+2	2.2E+3±7.1E+2	**1.4E+3±3.7E+2**	1.6E+3±4.4E+2	2.4E+4±8.9E+3	5.7E+3±1.1E+3	**4.8E+3±8.1E+2**
F13	**3.2E+0±2.4E+0**	3.6E+0±2.6E+0	3.4E+0±2.4E+0	1.6E+1±7.1E+0	**1.5E+1±5.9E+0**	**1.5E+1±4.7E+0**	6.4E+1±3.1E+1	4.9E+1±3.2E+1	**3.5E+1±1.5E+1**	1.2E+3±7.8E+2	**9.4E+1±3.3E+1**	1.3E+2±4.1E+1
F14	1.7E-1±3.6E-1	**2.0E-2±1.4E-1**	5.3E-2±2.0E-1	**2.2E+1±1.1E+0**	2.3E+1±3.3E+0	2.3E+1±1.8E+0	3.2E+1±4.0E+0	**2.7E+1±2.4E+0**	2.9E+1±3.9E+0	2.7E+2±2.7E+1	**5.4E+1±7.1E+0**	7.8E+1±1.4E+1
F15	**1.7E-1±2.1E-1**	2.7E-1±2.1E-1	4.1E-1±2.7E-1	3.8E+0±1.3E+0	**2.2E+0±1.2E+0**	5.4E+0±2.8E+0	4.5E+1±1.5E+1	2.4E+1±2.4E+0	**1.6E+1±6.7E+0**	2.6E+2±4.6E+1	**9.7E+1±3.6E+1**	1.1E+2±3.4E+1
F16	**4.1E-1±1.9E-1**	5.2E-1±2.4E-1	6.4E-1±2.3E-1	4.2E+1±3.3E+1	**3.1E+1±2.3E+1**	3.8E+1±5.5E+1	3.8E+2±1.4E+2	**3.0E+2±1.3E+2**	4.1E+2±1.7E+2	1.6E+3±2.1E+2	**1.4E+3±2.6E+2**	**1.4E+3±4.0E+2**
F17	1.7E-1±1.8E-1	**1.2E-1±2.0E-1**	1.5E+0±4.4E+0	3.3E+1±6.7E+0	**2.8E+1±6.2E+0**	3.2E+1±8.7E+0	2.5E+2±3.7E+1	**2.3E+2±6.6E+1**	2.9E+2±1.2E+2	1.1E+3±1.8E+2	1.1E+3±2.5E+2	**9.9E+2±3.4E+2**
F18	**2.8E-1±2.2E-1**	2.4E+0±6.0E+0	2.0E+0±5.4E+0	2.3E+1±2.0E+0	**2.1E+1±7.4E-1**	2.3E+1±2.0E+0	4.8E+1±1.7E+1	**2.5E+1±1.9E+0**	3.3E+1±7.2E+0	2.1E+2±3.8E+1	**9.8E+1±2.7E+1**	1.3E+2±2.4E+1
F19	**1.1E-2±1.1E-2**	5.5E-2±2.3E-1	8.1E-2±2.1E-1	6.1E+0±2.1E+0	**4.9E+0±1.7E+0**	9.4E+0±2.0E+0	3.4E+1±8.7E+0	**1.7E+1±2.6E+0**	2.2E+1±4.8E+0	1.8E+2±2.6E+1	**5.8E+1±9.0E+0**	7.4E+1±1.2E+1
F20	**1.5E-2±6.8E-2**	1.8E-1±1.6E-1	1.8E-1±1.7E-1	3.1E+1±8.6E+0	**2.8E+1±6.4E+0**	8.2E+1±5.9E+1	1.7E+2±6.8E+1	**1.3E+2±5.2E+1**	1.6E+2±1.1E+2	1.4E+3±2.0E+2	1.4E+3±1.8E+2	**1.3E+3±3.6E+2**
F21	1.6E+2±5.1E+1	1.6E+2±5.0E+1	**1.5E+2±4.9E+1**	**2.1E+2±1.5E+0**	**2.1E+2±1.9E+0**	**2.1E+2±4.2E+0**	2.2E+2±2.0E+0	2.3E+2±5.8E+0	**2.1E+2±9.2E+0**	2.6E+2±5.0E+0	2.7E+2±1.1E+1	**2.4E+2±1.1E+1**
F22	**1.0E+2±7.5E-2**	**1.0E+2±5.6E-2**	**1.0E+2±7.8E-2**	**1.0E+2±0.0E+0**	**1.0E+2±0.0E+0**	**1.0E+2±0.0E+0**	2.8E+3±1.3E+3	1.5E+3±1.7E+3	**8.2E+2±1.5E+3**	1.1E+4±4.1E+2	1.0E+4±5.9E+2	**9.0E+3±9.3E+2**
F23	**3.0E+2±9.0E-1**	**3.0E+2±1.5E+0**	**3.0E+2±1.5E+0**	3.6E+2±3.1E+0	3.6E+2±2.7E+0	3.6E+2±4.1E+0	**4.3E+2±3.4E+0**	4.4E+2±6.6E+0	4.4E+2±7.3E+0	**5.6E+2±7.1E+0**	5.9E+2±7.3E+0	5.8E+2±7.7E+0
F24	3.2E+2±5.1E+1	**2.9E+2±8.6E+1**	**2.9E+2±8.3E+1**	**4.3E+2±1.9E+0**	**4.3E+2±2.0E+0**	**4.3E+2±2.4E+0**	**5.1E+2±2.5E+0**	**5.1E+2±6.3E+0**	**5.1E+2±5.8E+0**	**9.2E+2±6.1E+0**	9.3E+2±2.1E+1	**9.2E+2±1.6E+1**
F25	**4.1E+2±2.1E+1**	4.2E+2±2.3E+1	4.3E+2±2.2E+1	3.9E+2±2.3E-2	3.9E+2±7.7E-3	**3.8E+2±1.2E-2**	4.8E+2±3.5E+0	4.8E+2±3.8E+0	**4.6E+2±8.5E+0**	7.5E+2±2.8E+1	6.9E+2±3.7E+1	**6.7E+2±4.3E+1**
F26	**3.0E+2±0.0E+0**	**3.0E+2±0.0E+0**	**3.0E+2±0.0E+0**	9.8E+2±4.3E+1	9.6E+2±4.2E+1	**9.4E+2±3.6E+1**	1.2E+3±3.7E+1	1.3E+3±6.8E+1	**1.1E+3±7.0E+1**	3.4E+3±9.2E+1	3.2E+3±2.4E+2	**3.1E+3±7.1E+1**
F27	**3.9E+2±1.5E-1**	**3.9E+2±1.1E+0**	**3.9E+2±3.1E+0**	5.1E+2±5.1E+0	5.1E+2±3.8E+0	**5.0E+2±1.4E+1**	5.4E+2±1.4E+1	5.3E+2±8.3E+0	**5.0E+2±9.0E-5**	6.6E+2±1.5E+1	5.9E+2±1.4E+1	**5.0E+2±1.3E-4**
F28	**3.6E+2±1.3E+2**	4.0E+2±1.5E+2	**3.6E+2±8.3E+1**	3.4E+2±5.6E+1	**3.2E+2±4.3E+1**	3.3E+2±5.1E+1	4.6E+2±1.4E+1	4.6E+2±9.6E+0	**4.4E+2±8.5E+0**	5.3E+2±2.7E+1	5.1E+2±1.5E+1	**5.0E+2±1.1E-4**
F29	**2.3E+2±2.0E+0**	**2.3E+2±2.6E+0**	**2.3E+2±2.2E+0**	4.4E+2±7.0E+0	4.3E+2±6.1E+0	**3.9E+2±4.1E+1**	3.5E+2±9.9E+0	3.5E+2±1.1E+1	**2.7E+2±6.0E+1**	1.4E+3±1.6E+2	**1.2E+3±1.5E+2**	1.3E+3±2.7E+2
F30	7.8E+4±2.5E+5	4.1E+4±2.1E+5	**2.3E+2±3.3E+1**	2.0E+3±6.8E+1	2.0E+3±7.3E+1	**4.0E+2±9.1E+1**	6.6E+5±1.1E+5	6.2E+5±5.3E+4	**6.5E+2±4.1E+2**	2.4E+3±1.7E+2	2.4E+3±2.0E+2	**5.6E+2±6.3E+1**