Continuous Parameter Pools in Ensemble Self-Adaptive Differential Evolution

Giovanni Iacca*, Fabio Caraffini†‡, and Ferrante Neri†‡
*INCAS^3, Dr. Nassaulaan 9, 9401 HJ Assen, The Netherlands
†Centre for Computational Intelligence, School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LE1 9BH, England, United Kingdom
‡Department of Mathematical Information Technology, P.O. Box 35 (Agora), 40014 University of Jyväskylä, Finland
Email: {giovanniiacca@incas3.eu, fabio.caraffini@dmu.ac.uk, fneri@dmu.ac.uk}

***Abstract*—Ensemble of parameters and mutation strategies differential evolution (EPSDE) is an elegant, promising optimization framework recently introduced in the literature. The idea behind it is that a pool of mutation and crossover strategies, along with associated pools of parameters, can flexibly adapt to a large variety of problems when a simple success based rule is introduced. Modern versions of this scheme attempts to improve upon the original performance at the cost of a high complexity. One of most successful implementations of this algorithmic scheme is the Self-adaptive Ensemble of Parameters and Strategies Differential Evolution (SaEPSDE). This paper operates on the SaEPSDE, reducing its complexity by identifying some algorithmic components that we experimentally show as possibly unnecessary. The result of this de-constructing operation is a novel algorithm implementation, here referred to as "j" Ensemble of Strategies Differential Evolution (jESDE). The proposed implementation is drastically simpler than SaEPSDE as several parts of it have been removed or simplified. Nonetheless, jESDE appears to display a competitive performance, on diverse problems throughout various dimensionality values, with respect to the original EPSDE algorithm, as well as to SaEPSDE and three modern algorithms based on Differential Evolution.**

I. Introduction

Differential Evolution (DE), see [1], is a popular metaheuristic that thanks to its simplicity and high performance on a wide spectrum of problems, has been successfully used in several fields of engineering and applied sciences. After its definition in 1995, see [2], DE has been applied to many domains. Some striking examples of applications of DE presented in the literature can be found, for instance, in sensor fusion [3], image processing (with reference to paper production) [4], and industrial robotics [5].

Over the past ten years, many research papers on DE have been focused on improved versions and implementations of the original scheme. As reported in [6], DE has a wide margin of improvement due to the fact that it employs a limited number of search moves. Although in some cases multiple variations are performed within the same implementations, modified DE schemes can be divided into the following categories:

- DE frameworks that include extra moving operators, such as multiple mutation operators or local search components, e.g. [7], [8], [9], [10] and [11];
- DE frameworks that employ a randomization of parameters or the search operators, e.g. [12], [13], [14], and [15].

Jointly to these variations in the search techniques, usually some adaptive schemes are included. These schemes are often based on the success of parameters or operators, such as in [16]. A successful example of such schemes is EPSDE [17]: in a nutshell, EPSDE is a variation of DE that encompasses multiple mutation/crossover operators and parameters, a randomization in the search, and a success-based adaptive logic (hence its name, i.e. Ensemble of Parameters and Strategies Differential Evolution). In turn, many variants of EPSDE have also been proposed in the literature. Some examples are the ensemble of compact DE, in [18], the ensemble of DE populations in [19], and constraint handling techniques in [20].

The main idea of the ensemble is simple and effective: parameter settings, search strategies, and operators are arranged in "pools", and selected in a randomized way. During an initial learning stage, the success of each choice biases the randomized selection mechanism so that the most successful choices have a higher chance to be selected. Despite its intuitiveness, the successful functioning of the algorithm as a whole can be a complex issue and may depend on the optimization problem under examination. Furthermore, there is currently no general knowledge about how to set the pools. This results in trial-and-error design processes which usually lead to very complex algorithm implementations, see [21]. Within the context of ensemble structures in DE frameworks, an example of effective implementation is reported in [22], where EPSDE is hybridized with the self-adaptive logic proposed in [7]. The resulting framework, named SaEPSDE, additionally incorporates as a local search the large-scale optimization method introduced in [23], which is in turn the combination of three optimizers.

This paper focuses on SaEPSDE, analysing its algorithmic functioning in the attempt to identify redundant (i.e., possibly unnecessary) components and thus obtain a new implementation which, albeit simpler, still displays a non-degraded performance. Furthermore, the implementation proposed here is realized in consideration of the fact that a pool of parameter settings may introduce a bias without a prior knowledge of the problem. On the contrary, we use a randomized sampling within continuous intervals in the fashion of the logic in [13].

The remainder of the paper is organized as follows. Section II shortly summarizes the working principles of DE, EPSDE, and SaEPSDE. Section III describes our proposed implementation. Section IV shows the numerical results in different

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.216

dimensionalities. Finally, Section V concludes this work.

II. Ensemble of Parameters and Mutation Differential Evolution

Without a loss of generality, this paper refers to the solution of the minimization of the function $f(\mathbf{x})$ with $\mathbf{x} = (x[1], x[2], \ldots x[n])$ in a $n-$dimensional decision space D.

Basic DE is a population-based metaheuristic where each individual in the population generates an offspring in two steps, mutation and crossover. Scanning the population individuals by running an index i from 1 to the population size S_{pop}, for each individual $\mathbf{x_i}$ a mutant individual $\mathbf{x_{off}}'$ is generated by applying a formula that contains at least one difference between two solutions (this features gives the name to the algorithm). The most classical mutation is the so-called DE/rand/1, which involves three individuals randomly selected from the population, $\mathbf{x_r}$, $\mathbf{x_s}$, and $\mathbf{x_t}$, and a constant parameter F. The mutant is obtained by applying the formula:

$$\mathbf{x_{off}}' = \mathbf{x_t} + F(\mathbf{x_r} - \mathbf{x_s}). \tag{1}$$

In the literature numerous other mutation formulas have been proposed, see [6]. The candidate solution $\mathbf{x_{off}}'$, often referred to as provisional offspring, is then hybridized with $\mathbf{x_i}$ by means of a crossover operator. Two possible crossovers are used in DE frameworks, namely binomial (abbreviated as "bin") and exponential ("exp").

The binomial crossover is defined as follows:

$$x_{off}[j] = \begin{cases} x_i[j] & \text{if } (rand(0,1) < CR) \text{ or } (j = j_{rand}) \\ x_{off}[j]' & \text{otherwise} \end{cases} \tag{2}$$

for $j = 1, 2, \ldots, n$. The crossover probability, $CR \in [0, 1]$, is a user-specified constant that controls the fraction of parameter values that are copied to the trial vector $\mathbf{x_{off}}$ (the offspring) from the mutant vector $\mathbf{x_{off}}'$. j_{rand} is a randomly chosen integer in the range $[1, n]$. The condition $(j = j_{rand})$ is introduced to ensure that the trial vector $\mathbf{x_{off}}$ will differ from its corresponding target vector $\mathbf{x_{off}}'$ by at least one parameter.

In exponential crossover, an integer j_{rand}, chosen randomly in $[1, n]$, acts as a starting point in the target vector, from where the exchange of components with the mutant vector starts. Subsequently, a set of random numbers between 0 and 1 are generated. As long as $rand(0,1) \leq CR$, where $CR \in [0, 1]$ is again a predetermined parameter, the design variables from the mutant vector are copied into the corresponding positions of the trial vector. The first time that $rand(0,1) > CR$, the copy process is interrupted. Then, all the remaining design variables of the trial vector are copied from the target vector. For the sake of clarity, the pseudo-code of the exponential crossover is shown in Algorithm 1.

When the offspring has been generated, its fitness value is calculated and compared with that of $\mathbf{x_i}$. The best of the solutions is then retained in the subsequent generation (this selection scheme is also known as one-to-one spawning, see [6]). The process goes on until a stop criterion is met. Starting from the original DE algorithm, the EPSDE scheme modifies it by including an implicit adaptation and multiple choices of mutation/crossover strategies and parameters. More specifically, EPSDE is associated to four pools:

Algorithm 1 Exponential crossover pseudo-code

```
x_off = x_i
generate j = j_rand
x_off[j] = x_off[j]'
while rand(0,1) ≤ CR do
  j = j + 1
  if j == n then
    j = 1
  end if
  x_off[j] = x_off[j]'
end while
```

- pool of scale factors F (P_F)
- pool of crossover rates CR (P_{CR})
- pool of mutation strategies (P_{mut})
- pool of crossover strategies (P_{cross})

The first two pools contain the operator parameters that can be selected. The latter two are more of interest. With reference to [17], the mutation strategies in the pool are:

- DE/cur-to-rand/1 $\mathbf{x_{off}}' = \mathbf{x_i} + K(\mathbf{x_r} - \mathbf{x_i}) + F(\mathbf{x_s} - \mathbf{x_t})$
- DE/cur-to-pbest/1 $\mathbf{x_{off}}' = \mathbf{x_i} + F(\mathbf{x^p_{best}} - \mathbf{x_i}) + F(\mathbf{x_s} - \mathbf{x_t})$

where $\mathbf{x^p_{best}}$ is randomly chosen as one of the top $100p\%$ individuals in the current population, with $p \in (0, 1]$.

The last pool contains the two possible crossovers, i.e. bin and exp as previously described.

It should be observed that after the DE/cur-to-rand/1 mutation, the crossover operation is not applied, as this mutation strategy contains an implicit arithmetic crossover, see [24]. Conversely, a crossover is applied after DE/cur-to-pbest/1 strategy. Furthermore, DE/cur-to-pbest/1 makes use of a dynamic p value, according to the following rule:

$$p = \lfloor 0.005 \cdot (1 - n_{eval}/N_{eval}) \rfloor \tag{3}$$

where n_{eval} and N_{eval} indicate, respectively, the current and maximum number of fitness evaluation. In this way, at the beginning of the optimization ($n_{eval} = 0$) the mutation uses a random solution among the top 50% individuals in the current population: this guarantees a higher chance of taking a suboptimal solution, thus increasing the exploration pressure. Later on ($n_{eval} \to N_{eval}$), the percentage of top individuals will progressively decrease, thus making the mutation strategy more exploitative.

The selection and coordination of the parameters and operators (mutation and crossover) is performed in a simplistic, elegant, and effective manner. A setting of two parameters and two strategies is randomly assigned to each individual, i.e. for each individual a setting (i.e., a tuple of strategies and parameters) is sampled from the pools $\{P_{mut}, P_{cross}, P_F, P_{CR}\}$. If the setting is successful at generating an offspring that outperforms the parent $\mathbf{x_i}$, the strategies/parameter setting is retained and inherited by the offspring solution. If the setting does not lead to the generation of a successful offspring, a new setting is randomly sampled.

Algorithm 2 shows the pseudo-code of the implementation of EPSDE algorithm proposed in [17].

Algorithm 2 EPSDE pseudo-code

```
initialize a pool of mutation strategies P_mut and crossover strategies P_cross
initialize a pool of scale factors P_F and crossover probabilities P_CR
generate N_p individuals of the initial population pseudo-randomly
for i = 1 : N_p do
    assign to x_i random strategies/parameters from {P_mut, P_cross, P_F, P_CR}
    compute f(x_i)
end for
while budget condition do
    for i = 1 : N_p do
        generate x'_off through mutation strategy/parameter associated to x_i
        generate x_off through crossover strategy/parameter associated to x_i
        if f(x_off) ⩽ f(x_i) then
            save index i for replacing x_i = x_off (including mutation and crossover strategies as well as parameters) in the next generation
        else
            assign to x_i new random strategies/parameters from the pools
        end if
    end for
    perform replacements
end while
```

In order to enhance the performance of the EPSDE scheme, a self-adaptive version has been proposed by incorporating within it the self-adaptation proposed in [7], thus generating Self-adaptive EPSDE (SaEPSDE). The main additions to the original EPSDE scheme are that a learning period is performed at the beginning of the optimization and a population split is performed in the first 70% of the computational budget. At the beginning of the optimization process, four pools of strategies/parameters are set like in EPSDE and one setting is associated to each individual. Since the beginning of the optimization and for a period here referred to as Splitting Period (and consisting of SP fitness evaluations), the population, at each generation, is structured into two subpopulations. The first one is composed of the N_{ps} individuals with the worst fitness. The subpopulation size N_{ps} is given by:

$$N_{ps} = round\left(\frac{N_p}{2} + \frac{N_p}{2}\frac{n_{eval}}{SP}\right) \tag{4}$$

where n_{eval} is the budget spent and N_{eval} is the total budget in terms of fitness evaluations. The remaining individuals with the best fitness values compose the second subpopulation.

At the beginning of the optimization, for a budget of LP fitness evaluations, the learning period is performed. During this period, EPSDE is performed on the first subpopulation after having assigned to each individual a random setting (mutation, crossover, F, and CR). During the learning period the success of strategies and parameters is monitored and, for each of them, the success rate is recorded. The remaining $N_p - N_{ps}$ individuals evolve like a simplified EPSDE. More specifically, the DE/cur-to-rand/1 mutation shown above is used. In this case, $K = 0.5\,(1 + F_i)$ and F_i is sampled from the EPSDE pool and gets updated according the EPSDE success rule. As for crossover, these individuals undergo only exponential crossover with a fixed crossover rate $CR = 0.9$.

At the end of the learning period the most successful setting is used for the first subpopulation while the remaining individuals (second subpopulation) are processed with DE/cur-to-rand/1 mutation and exponential crossover as described above (as during the learning period).

At the end of the splitting period and until the budget exhaustion, the entire population evolves as a DE with the most successful setting detected during the learning period. The working principles of SaEPSDE are illustrated in Algorithm 3.

III. j Ensemble of Strategies Differential Evolution

The SaEPSDE scheme, albeit capable to perform very well on various optimization problems, is clearly complex as it is composed of several parts, see Algorithm 3. Furthermore, the design appears mostly empirical while the role of each component is not fully clear.

We introduce here a simplified version of SaEPSDE with the aim of not compromising on the performance of the algorithm. More specifically, starting from SaEPSDE, we focus on some of its components that in our view could be either simplified or removed at all. In a way, this work can be seen as an algorithm deconstruction aiming at highlighting the components that are effective in the search. The resulting algorithm, here indicated as "j" Ensemble of Strategies Differential Evolution (jESDE), is described in the following.

At the beginning of the optimization, as usual N_p solutions are sampled within the decision space. Along with each solution $\mathbf{x_i}$, a scale factor F_i and a crossover rate CR_i, sampled with a uniform distribution from the *continuous* intervals $(0, 1.2]$ and $(0, 1)$, respectively, are associated. Furthermore, a pool of mutation and crossover strategies is also set in the EPSDE fashion. As for the mutation strategies we used two straightforward ones, namely the classical DE/rand/1, as per eq (1), and DE/cur-to-best:

$$\mathbf{x_{off}}' = \mathbf{x_i} + F\,(\mathbf{x_{best}} - \mathbf{x_i}) + F\,(\mathbf{x_s} - \mathbf{x_t})\,. \tag{5}$$

Moreover, the crossover strategies are bin and exp, just like in EPSDE and SaEPSDE.

Similar to SaEPSDE, the proposed jESDE performs a learning on the mutation and crossover strategies. The learning mechanism is the one used in SaEPSDE, and at the end of the learning period the most successful strategies are detected and used on all the individuals of the population till the end of the optimization process. As for F and CR instead, rather than learning on two finite pools of values as SaEPSDE does, jESDE applies throughout the entire evolution the controlled randomization of parameters used in jDE [13] (hence the "j" in the algorithm name). More specifically, the scale factor F_i and the crossover rate CR_i associated to each individual are

Algorithm 3 SaEPSDE pseudo-code

initialize a pool of mutation strategies P_{mut} and crossover strategies P_{cross}
initialize a pool of scale factors P_F and crossover probabilities P_{CR}
generate N_p individuals of the initial population pseudo-randomly
for $i = 1 : N_p$ **do**
  assign to $\mathbf{x_i}$ random strategies/parameters from $\{P_{mut}, P_{cross}, P_F, P_{CR}\}$
  compute $f(\mathbf{x_i})$
end for
while budget condition on N_{eval} **do**
  if the splitting period is on $n_{eval} \leq SP$ **then**
    calculate the first subpopulation size $N_{ps} = round\left(\frac{N_p}{2} + \frac{N_p}{2}\frac{n_{eval}}{SP}\right)$
    if the learning period is on $n_{eval} \leq LP$ **then**
      for $i = 1 : N_{ps}$ **do**
        generate $\mathbf{x'_{off}}$ through mutation strategy/parameter associated to $\mathbf{x_i}$
        generate $\mathbf{x_{off}}$ through crossover strategy/parameter associated to $\mathbf{x_i}$
        if $f(\mathbf{x_{off}}) \leqslant f(\mathbf{x_i})$ **then**
          update the successful strategy registry
          save index i for replacing $\mathbf{x_i} = \mathbf{x_{off}}$ (including mutation and crossover strategies as well as parameters) in the next generation
        else
          assign to $\mathbf{x_i}$ new random strategies/parameters from the pools
        end if
      end for
      perform replacements
      for $i = N_{ps} + 1 : N_p$ **do**
        apply DE/cur-to-rand/1 mutation and exponential crossover to generate $\mathbf{x_{off}}$
        if $f(\mathbf{x_{off}}) \leqslant f(\mathbf{x_i})$ **then**
          update the successful strategy registry
          save index i for replacing $\mathbf{x_i} = \mathbf{x_{off}}$ (including mutation and crossover strategies as well as parameters) in the next generation
        else
          assign to $\mathbf{x_i}$ new random strategies/parameters from the pools
        end if
      end for
      perform replacements
    else
      for $i = 1 : N_{ps}$ **do**
        apply DE with the best setting found during the learning period
      end for
      perform replacements
      for $i = N_{ps} + 1 : N_p$ **do**
        apply DE/cur-to-rand/1 mutation with and exponential crossover
        compare performance and update the success of F_i
      end for
      perform replacements
    end if
  else
    for $i = 1 : N_p$ **do**
      apply DE with the best setting found during the learning period
    end for
    perform replacements
  end if
end while

updated according to the following rules:

$$F_i = \begin{cases} F_l + F_u rand_1, & \text{if } rand_2 < \tau_1 \\ F_i, & \text{otherwise} \end{cases} \quad (6)$$

$$CR_i = \begin{cases} rand_3, & \text{if } rand_4 < \tau_2 \\ CR_i, & \text{otherwise} \end{cases} \quad (7)$$

where $rand_j$, $j \in \{1, 2, 3, 4\}$, are uniform pseudo-random values between 0 and 1; τ_1 and τ_2 are constant values which represent the probabilities that parameters are updated, F_l and F_u are constant values which represent the minimum value that F could take and the maximum variable contribution to F, respectively. The newly calculated values of F_i and CR_i

are then used for generating the offspring with the proper mutation and crossover strategies chosen from the pools. For the sake of completeness, we report the pseudo-code of jESDE in Algorithm 4.

Algorithm 4 jESDE pseudo-code

```
initialize a pool of mutation strategies P_mut and crossover strategies P_cross
generate N_p individuals of the initial population pseudo-randomly
for i = 1 : N_p do
  assign to x_i random strategies from {P_mut, P_cross}
  assign to x_i F_i and CR_i sampled from continuous intervals
  compute f(x_i)
end for
while budget condition on N_eval do
  for i = 1 : N_p do
    update F_i and CR_i as per eq (6) and (7)
  end for
  if the learning period is on n_eval ≤ LP then
    for i = 1 : N_p do
      generate x'_off through mutation strategy/parameter associated to x_i
      generate x_off through crossover strategy/parameter associated to x_i
      if f(x_off) ⩽ f(x_i) then
        update the successful strategy registry
        save index i for replacing x_i = x_off (including mutation and crossover strategies) in the next generation
      else
        assign to x_i new random strategies from the pools
      end if
    end for
    perform replacements
  else
    for i = 1 : N_ps do
      apply jDE with the best strategies found during the learning period
    end for
    perform replacements
  end if
end while
```

A first thing to note about the proposed jESDE is that it keeps the learning period structure of SaEPSDE, as it appears to have a major impact on the performance of the algorithm. On the contrary, we observed that the splitting period does not have a dramatic effect on the optimization results.

Another fundamental point has to do with the use of multiple and diverse mutation operators. As shown by our experiments (reported below), we conjecture that effective mutation strategies should be applied for a certain portion of the computational budget in order to exploit efficiently the search directions (in a way, mutations require some "learning"). However, given a fixed budget, the use of too many (and too complex) mutation strategies might not necessarily be effective. To our understanding, the employment of two simple mutation strategies is usually enough to have the benefits of multiple operators, as each of them is allocated a sufficient number of fitness evaluations to produce effective improvements. Intuitively, the two mutation operators should be diverse and somehow complementary: as for the use of multiple local search operators in Memetic Algorithms, see [25] and [26], the design of mutation strategies in a DE framework should attempt to balance global and local search. In the specific case of jESDE, we have chosen the DE/rand/1 that is highly randomized, hence fairly exploratory and with global properties, and the DE/cur-to-best that is quite exploitative as it performs a search around the best individual of the population. It must be remarked that the two mutation strategies selected are fairly simple since they do not need to rank the solutions (as instead happens in SaEPSDE). This choice has been done purposely, in order to simplify the original scheme.

Finally, the jDE controlled randomization of [13] appears to very often improve upon the performance of a DE scheme. In this case, a fixed number of possible parameters can still offer not enough search moves to the DE framework, see [6] and[27]. On the contrary, this randomization appears to offer a good balance as it increases the pool of search moves without making the algorithm be excessively exploratory.

IV. Numerical Results

The proposed jESDE has been tested with the following parameters: population size $N_p = 50$, $F_l = 0.1$, $F_u = 0.9$, $\tau_1 = \tau_2 = 0.1$ and learning period $LP = 20$. It should be noted that no specific parameter tuning was performed, as we preferred to use the original parameters of jDE and SADE (see below) in order to obtain a fair comparison.

The jESDE algorithm has been then tested against the original EPSDE [17] and SaEPSDE [22], the latter being the algorithmic scheme that has been the starting point for this algorithmic simplification work. The following parameter setting, as suggested in the original papers, has been used for these algorithms:

- EPSDE [17]: population size $N_p = 50$, and parameter pools $P_F = \{0.5, 0.9\}$ and $P_{CR} = \{0.1, 0.5, 0.9\}$;
- SaEPSDE [22]: same N_p, P_F and P_{CR} as for EPSDE, learning period $LP = 20$ and splitting period $SP = 70\%\ N_{eval}$.

Furthermore, the following DE-based state-of-the-art algorithms have been included in this study, using the suggested parameter setting:

- MDe-pBX [28]: $N_p = 100$ and group size $q = 15$;
- jDE [13]: $N_p = 50$, $F_l = 0.1$, $F_u = 0.9$, and $\tau_1 = \tau_2 = 0.1$;
- SaDE [7]: $N_p = 50$ and learning period $LP = 20$.

These algorithms have been tested over the CEC2013 testbed [29] in 10, 30, and 50 dimensions and over the CEC2010 testbed [30] in 1000 dimensions. Each algorithm has been run for $5000 \times n$ (being n the problem dimension) fitness evaluations for each problem of the testbeds. Each single experiment has been repeated for 100 independent runs. All the pairwise comparisons have been statistically checked by means of the Wilcoxon test [31], while a statistical ranking

TABLE VI. HOLM-BONFERRONI PROCEDURE (W.R.T. JESDE, RANK = 4.27). RANK IS THE AVERAGE SCORE OF THE ALGORITHM THROUGHOUT ALL BENCHMARKS (5 IS BEST), z_j, p_j AND δ/j ARE DEFINED AS IN [32]. "ACCEPTED" INDICATES STATISTICAL EQUIVALENCE, "REJECTED" INDICATES SUPERIORITY OF JESDE.

j	Optimizer	Rank	z_j	p_j	δ/j	Hypothesis
1	jDE	4.14	-5.70e-01	2.84e-01	5.00e-02	Accepted
2	SaEPSDE	3.51	-3.46e+00	2.66e-04	2.50e-02	Rejected
3	EPSDE	3.03	-5.66e+00	7.70e-09	1.67e-02	Rejected
4	SADE	2.90	-6.23e+00	2.38e-10	1.25e-02	Rejected
5	MDE-pBX	2.61	-7.59e+00	1.64e-14	1.00e-02	Rejected

has been performed by Holm-Bonferroni procedure, see [32]. In all tests we assumed a confidence level of 0.95 ($\alpha = 0.05$).

Numerical results for CEC2013 are displayed in Tables I, II, III, and IV (for brevity we omit the detailed results in 10 dimensions). For each problem and algorithm, we show the average error and standard deviation w.r.t. the known optimal fitness value. In the tables, a "+" indicates that jESDE statistically outperforms its competitor, a "-" indicates that jESDE is outperformed, while "=" 'indicates that the algorithms are statistically equivalent. The results on CEC2013 throughout the available dimensionality values clearly show that the proposed jESDE appears to promisingly tackle most of -if not all- the benchmark problems considered in the experiments. As for CEC2010, detailed results of jESDE against MDE-pBX, jDE and SADE are shown in Table V, with the same notation used in the previous tables. The other detailed results in 1000 dimensions are omitted for space restrictions.

Finally, the results of the Holm-Bonferroni procedure related to all the results calculated are reported in Table VI. It can be observed that the proposed jESDE displays the best performance with respect to the other five algorithms over the 104 problems considered in this study. However, as seen in Table V the performance of jESDE gets deteriorated in high dimensions while jDE proves to offer a very robust performance throughout the various dimensionality values. To our understanding, large-scale problems require a high exploitation within the given limited budget. It must be remarked that while in common optimization practice the budget grows linearly with the problem dimensionality ($5000 \times n$), the search space grows exponentially, making de facto large-scale runs way shorter than low-dimensional runs. Given these circumstances, we believe that DE implementations that use only one mutation/crossover strategy and rapidly exploit the search directions are likely to be more successful in high dimensions than those that use multiple search operators.

V. CONCLUSION

This paper proposed a novel, self-adaptive implementation of the EPSDE algorithm, here indicated as jESDE. The proposed algorithm makes use of two simple mutation and two crossover strategies. The adaptation of the strategies is achieved by a learning period and a straightforward implementation of the ensemble mechanism. Furthermore, jESDE uses the controlled randomization of F and CR proposed in the jDE framework. The resulting algorithm displays an excellent performance in relatively low dimensions and a moderately good performance in high dimensions. In the latter case though, the base jDE algorithm shows the best results.

The de-constructing task performed to obtain jESDE led us to draw some conclusions regarding the DE functioning: 1) the learning period, albeit short, supports DE to adapt to the problem and appears to help to detect better solutions; 2) multiple mutations/crossovers are useful to tackle diverse problems; 3) the mutation operators must be diverse to work properly, e.g. one should have exploratory features while the other should be more exploitative; 4) complex operators do not necessarily enhance the performance of a DE algorithm, especially when diverse optimization problems are considered.

REFERENCES

[1] S. Das and P. Suganthan, "Differential Evolution: A Survey of the State-of-the-Art," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 1, pp. 4–31, 2011.

[2] R. Storn and K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," TR-95-012, 1995.

[3] R. Joshi and A. C. Sanderson, "Minimal representation multisensor fusion using differential evolution," *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 29, no. 1, pp. 63–76, 1999.

[4] V. Tirronen, F. Neri, T. Kärkkäinen, K. Majava, and T. Rossi, "An enhanced memetic differential evolution in filter design for defect detection in paper production," *Evolutionary Computation*, vol. 16, pp. 529–555, 2008.

[5] F. Neri and E. Mininno, "Memetic Compact Differential Evolution for Cartesian Robot Control," *IEEE Computational Intelligence Magazine*, vol. 5, no. 2, pp. 54–65, 2010.

[6] F. Neri and V. Tirronen, "Recent Advances in Differential Evolution: A Review and Experimental Analysis," *Artificial Intelligence Review*, vol. 33, no. 1–2, pp. 61–106, 2010.

[7] 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.

[8] J. Zhang and A. Sanderson, "JADE: Adaptive Differential Evolution With Optional External Archive," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 5, pp. 945–958, 2009.

[9] F. Neri and V. Tirronen, "Scale Factor Local Search in Differential Evolution," *Memetic Computing Journal*, vol. 1, no. 2, pp. 153–171, 2009.

[10] N. Noman and H. Iba, "Accelerating Differential Evolution Using an Adaptive Local Search," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 107–125, 2008.

[11] G. Iacca, F. Caraffini, and F. Neri, "Multi-Strategy Coevolving Aging Particle Optimization," *International Journal of Neural Systems*, vol. 24, no. 01, p. 1450008, 2014.

[12] S. Das and A. Konar, "An improved differential evolution scheme for noisy optimization problems," in *Pattern recognition and machine intelligence*, ser. LNCS. Springer, 2005, vol. 3776, pp. 417–421.

[13] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, "Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[14] E. Mininno, F. Neri, F. Cupertino, and D. Naso, "Compact Differential Evolution," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 32–54, 2011.

[15] F. Neri, G. Iacca, and E. Mininno, "Disturbed Exploitation compact Differential Evolution for Limited Memory Optimization Problems," *Information Sciences*, vol. 181, no. 12, pp. 2469–2487, 2011.

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

[17] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing*, vol. 11, no. 2, pp. 1679–1696, 2011.

TABLE I. AVERAGE ERROR ± STANDARD DEVIATION AND STATISTIC COMPARISON (REFERENCE: JESDE) FOR JESDE AGAINST EPSDE AND SAEPSDE ON CEC2013 [29] IN 30 DIMENSIONS. THE BOLD FONT INDICATES THE ALGORITHM WITH THE LOWEST AVERAGE ERROR.

	jESDE	EPSDE		SA-EPSDE	
f_1	$\mathbf{0.00e+00 \pm 0.00e+00}$	$\mathbf{0.00e+00 \pm 0.00e+00}$	=	$0.00e+00 \pm 3.78e-13$	+
f_2	$\mathbf{4.53e+04 \pm 2.78e+04}$	$3.36e+06 \pm 2.02e+06$	+	$2.56e+05 \pm 1.50e+05$	+
f_3	$6.70e+06 \pm 9.19e+06$	$\mathbf{1.24e+06 \pm 3.24e+06}$	-	$7.88e+07 \pm 1.54e+08$	+
f_4	$\mathbf{2.86e+00 \pm 3.12e+00}$	$1.88e+04 \pm 4.28e+03$	+	$1.22e+02 \pm 1.73e+02$	+
f_5	$0.00e+00 \pm 3.41e-14$	$\mathbf{0.00e+00 \pm 2.54e-14}$	=	$1.14e-13 \pm 2.91e-13$	+
f_6	$\mathbf{4.91e+00 \pm 5.03e+00}$	$1.43e+01 \pm 7.45e+00$	+	$1.98e+01 \pm 2.52e+01$	+
f_7	$4.27e+01 \pm 1.51e+01$	$\mathbf{3.27e+01 \pm 1.02e+01}$	-	$5.27e+01 \pm 1.38e+01$	+
f_8	$\mathbf{2.09e+01 \pm 5.31e-02}$	$2.10e+01 \pm 5.51e-02$	+	$2.10e+01 \pm 5.10e-02$	+
f_9	$\mathbf{1.90e+01 \pm 3.48e+00}$	$3.19e+01 \pm 1.33e+00$	+	$2.21e+01 \pm 3.18e+00$	+
f_{10}	$8.50e-02 \pm 5.78e-02$	$\mathbf{3.27e-02 \pm 1.90e-02}$	-	$9.67e-02 \pm 8.50e-02$	=
f_{11}	$1.99e-02 \pm 1.39e-01$	$\mathbf{2.22e-04 \pm 5.16e-04}$	+	$2.07e+01 \pm 1.03e+01$	+
f_{12}	$\mathbf{6.85e+01 \pm 1.60e+01}$	$1.07e+02 \pm 1.22e+01$	+	$9.18e+01 \pm 2.30e+01$	+
f_{13}	$\mathbf{1.27e+02 \pm 2.53e+01}$	$1.29e+02 \pm 1.42e+01$	=	$1.55e+02 \pm 2.88e+01$	+
f_{14}	$1.28e+02 \pm 3.57e+01$	$6.87e+02 \pm 1.20e+02$	+	$\mathbf{3.13e+01 \pm 2.79e+01}$	-
f_{15}	$\mathbf{3.54e+03 \pm 7.20e+02}$	$6.25e+03 \pm 3.16e+02$	+	$3.81e+03 \pm 7.58e+02$	+
f_{16}	$\mathbf{1.79e+00 \pm 5.24e-01}$	$2.46e+00 \pm 3.13e-01$	+	$2.03e+00 \pm 4.02e-01$	+
f_{17}	$\mathbf{3.25e+01 \pm 3.45e-01}$	$3.27e+01 \pm 4.74e-01$	+	$3.32e+01 \pm 9.77e-01$	+
f_{18}	$\mathbf{8.37e+01 \pm 1.71e+01}$	$1.80e+02 \pm 1.34e+01$	+	$9.21e+01 \pm 2.05e+01$	+
f_{19}	$\mathbf{2.14e+00 \pm 5.79e-01}$	$3.23e+00 \pm 2.58e-01$	+	$3.78e+00 \pm 1.41e+00$	+
f_{20}	$\mathbf{1.04e+01 \pm 7.06e-01}$	$1.23e+01 \pm 2.73e-01$	+	$1.06e+01 \pm 6.31e-01$	+
f_{21}	$3.22e+02 \pm 7.32e+01$	$\mathbf{2.98e+02 \pm 8.49e+01}$	-	$3.28e+02 \pm 8.13e+01$	+
f_{22}	$2.73e+02 \pm 8.63e+01$	$1.05e+03 \pm 1.94e+02$	+	$\mathbf{1.58e+02 \pm 5.32e+01}$	-
f_{23}	$\mathbf{3.93e+03 \pm 7.70e+02}$	$6.40e+03 \pm 3.86e+02$	+	$4.28e+03 \pm 7.29e+02$	+
f_{24}	$\mathbf{2.31e+02 \pm 8.13e+00}$	$2.80e+02 \pm 5.22e+00$	+	$2.47e+02 \pm 1.14e+01$	+
f_{25}	$\mathbf{2.73e+02 \pm 1.41e+01}$	$2.91e+02 \pm 5.85e+00$	+	$2.87e+02 \pm 9.40e+00$	+
f_{26}	$\mathbf{2.00e+02 \pm 1.59e-03}$	$2.15e+02 \pm 4.90e+01$	+	$2.03e+02 \pm 1.94e+01$	+
f_{27}	$\mathbf{6.82e+02 \pm 9.82e+01}$	$1.10e+03 \pm 4.06e+01$	+	$7.39e+02 \pm 8.51e+01$	+
f_{28}	$2.96e+02 \pm 2.80e+01$	$3.00e+02 \pm 4.13e-13$	-	$\mathbf{2.88e+02 \pm 4.75e+01}$	+

TABLE II. AVERAGE ERROR ± STANDARD DEVIATION AND STATISTIC COMPARISON (REFERENCE: JESDE) FOR JESDE AGAINST EPSDE AND SAEPSDE ON CEC2013 [29] IN 50 DIMENSIONS. THE BOLD FONT INDICATES THE ALGORITHM WITH THE LOWEST AVERAGE ERROR.

	jESDE	EPSDE		SA-EPSDE	
f_1	$\mathbf{0.00e+00 \pm 0.00e+00}$	$\mathbf{0.00e+00 \pm 0.00e+00}$	=	$2.27e-13 \pm 7.07e-13$	+
f_2	$\mathbf{2.77e+05 \pm 9.88e+04}$	$1.30e+07 \pm 4.30e+06$	+	$8.42e+05 \pm 3.25e+05$	+
f_3	$5.45e+07 \pm 6.17e+07$	$\mathbf{1.20e+07 \pm 2.58e+07}$	-	$1.79e+08 \pm 1.61e+08$	+
f_4	$\mathbf{1.27e+01 \pm 1.24e+01}$	$5.15e+04 \pm 8.60e+03$	+	$3.01e+02 \pm 2.30e+02$	+
f_5	$\mathbf{1.14e-13 \pm 1.14e-14}$	$1.14e-13 \pm 1.14e-14$	=	$1.36e-12 \pm 1.15e-11$	+
f_6	$4.73e+01 \pm 1.76e+01$	$\mathbf{4.48e+01 \pm 1.27e+00}$	+	$5.71e+01 \pm 2.26e+01$	+
f_7	$6.63e+01 \pm 1.22e+01$	$8.51e+01 \pm 1.63e+01$	+	$\mathbf{5.87e+01 \pm 1.08e+01}$	-
f_8	$\mathbf{2.11e+01 \pm 3.81e-02}$	$2.11e+01 \pm 3.03e-02$	+	$2.12e+01 \pm 3.65e-02$	+
f_9	$4.45e+01 \pm 5.49e+00$	$6.09e+01 \pm 1.83e+00$	+	$\mathbf{4.25e+01 \pm 4.93e+00}$	-
f_{10}	$9.63e-02 \pm 5.16e-02$	$9.60e-02 \pm 4.92e-02$	=	$\mathbf{6.69e-02 \pm 1.19e-01}$	-
f_{11}	$3.58e-01 \pm 1.89e+00$	$\mathbf{2.95e-02 \pm 1.04e-01}$	+	$4.09e+01 \pm 1.58e+01$	+
f_{12}	$1.96e+02 \pm 4.28e+01$	$2.33e+02 \pm 2.35e+01$	+	$\mathbf{1.90e+02 \pm 2.99e+01}$	=
f_{13}	$3.16e+02 \pm 4.81e+01$	$\mathbf{2.89e+02 \pm 2.32e+01}$	-	$3.05e+02 \pm 4.28e+01$	=
f_{14}	$2.42e+02 \pm 8.16e+01$	$9.76e+02 \pm 1.50e+02$	+	$\mathbf{7.94e+01 \pm 5.46e+01}$	-
f_{15}	$\mathbf{7.59e+03 \pm 1.39e+03}$	$1.19e+04 \pm 5.06e+02$	+	$1.02e+04 \pm 1.82e+03$	+
f_{16}	$\mathbf{2.31e+00 \pm 5.52e-01}$	$3.13e+00 \pm 2.93e-01$	+	$3.23e+00 \pm 3.33e-01$	+
f_{17}	$\mathbf{5.58e+01 \pm 6.19e-01}$	$5.68e+01 \pm 7.48e-01$	+	$5.97e+01 \pm 2.71e+00$	+
f_{18}	$\mathbf{2.07e+02 \pm 3.94e+01}$	$3.43e+02 \pm 1.94e+01$	+	$2.69e+02 \pm 7.43e+01$	+
f_{19}	$6.46e+00 \pm 2.03e+00$	$\mathbf{5.85e+00 \pm 4.79e-01}$	=	$7.98e+00 \pm 2.24e+00$	+
f_{20}	$\mathbf{1.93e+01 \pm 8.29e-01}$	$2.20e+01 \pm 3.30e-01$	+	$2.01e+01 \pm 9.66e-01$	+
f_{21}	$8.22e+02 \pm 3.61e+02$	$\mathbf{6.24e+02 \pm 4.33e+02}$	-	$8.60e+02 \pm 3.25e+02$	+
f_{22}	$3.94e+02 \pm 1.87e+02$	$1.29e+03 \pm 2.85e+02$	+	$\mathbf{3.15e+02 \pm 2.68e+02}$	-
f_{23}	$\mathbf{8.17e+03 \pm 1.20e+03}$	$1.20e+04 \pm 6.26e+02$	+	$1.07e+04 \pm 2.04e+03$	+
f_{24}	$\mathbf{2.88e+02 \pm 1.25e+01}$	$3.54e+02 \pm 7.08e+00$	+	$2.96e+02 \pm 1.30e+01$	+
f_{25}	$\mathbf{3.64e+02 \pm 1.19e+01}$	$3.80e+02 \pm 6.12e+00$	+	$3.67e+02 \pm 1.26e+01$	=
f_{26}	$2.73e+02 \pm 9.61e+01$	$4.14e+02 \pm 8.91e+01$	+	$\mathbf{2.00e+02 \pm 3.49e-02}$	+
f_{27}	$\mathbf{1.32e+03 \pm 1.20e+02}$	$1.84e+03 \pm 5.88e+01$	+	$1.35e+03 \pm 1.17e+02$	+
f_{28}	$\mathbf{5.36e+02 \pm 6.69e+02}$	$7.15e+02 \pm 9.44e+02$	-	$6.46e+02 \pm 8.98e+02$	+

TABLE III. AVERAGE ERROR ± STANDARD DEVIATION AND STATISTIC COMPARISON (REFERENCE: JEPDE) FOR JEPDE AGAINST MDE-PBX, JDE, AND SADE ON CEC2013 [29] IN 30 DIMENSIONS. THE BOLD FONT INDICATES THE ALGORITHM WITH THE LOWEST AVERAGE ERROR.

	jESDE	MDE-pBX		jDE		SADE	
f_1	$\mathbf{0.00e+00 \pm 0.00e+00}$	$2.27e-13 \pm 4.86e-13$	+	$0.00e+00 \pm 3.94e-14$	=	$0.00e+00 \pm 8.51e-14$	+
f_2	$\mathbf{4.53e+04 \pm 2.78e+04}$	$2.70e+05 \pm 2.62e+05$	+	$5.90e+05 \pm 3.59e+05$	+	$7.54e+05 \pm 4.40e+05$	+
f_3	$6.70e+06 \pm 9.19e+06$	$5.19e+07 \pm 1.18e+08$	+	$\mathbf{9.25e+05 \pm 1.81e+06}$	-	$7.93e+07 \pm 1.28e+08$	+
f_4	$\mathbf{2.86e+00 \pm 3.12e+00}$	$3.49e+02 \pm 3.18e+02$	+	$5.53e+02 \pm 3.13e+02$	+	$1.32e+04 \pm 5.13e+03$	+
f_5	$\mathbf{0.00e+00 \pm 3.41e-14}$	$1.09e-10 \pm 1.00e-09$	+	$1.14e-13 \pm 2.78e-14$	+	$1.14e-13 \pm 1.22e-13$	+
f_6	$\mathbf{4.91e+00 \pm 5.03e+00}$	$3.41e+01 \pm 2.77e+01$	+	$9.68e+00 \pm 5.34e+00$	+	$2.16e+01 \pm 1.85e+01$	+
f_7	$4.27e+01 \pm 1.51e+01$	$5.61e+01 \pm 1.90e+01$	+	$\mathbf{2.56e+00 \pm 2.87e+00}$	-	$5.06e+01 \pm 1.70e+01$	+
f_8	$\mathbf{2.09e+01 \pm 5.31e-02}$	$2.10e+01 \pm 5.93e-02$	+	$2.10e+01 \pm 4.48e-02$	=	$2.10e+01 \pm 9.43e-02$	+
f_9	$\mathbf{1.90e+01 \pm 3.48e+00}$	$2.16e+01 \pm 4.36e+00$	+	$3.04e+01 \pm 4.00e+00$	+	$2.85e+01 \pm 3.49e+00$	+
f_{10}	$8.50e-02 \pm 5.78e-02$	$1.81e-01 \pm 1.10e-01$	+	$\mathbf{4.12e-02 \pm 2.33e-02}$	-	$1.55e-01 \pm 1.23e-01$	+
f_{11}	$1.99e-02 \pm 1.39e-01$	$4.68e+01 \pm 1.54e+01$	+	$\mathbf{0.00e+00 \pm 1.14e-14}$	=	$2.51e+01 \pm 1.19e+01$	+
f_{12}	$6.85e+01 \pm 1.60e+01$	$6.91e+01 \pm 2.20e+01$	=	$8.00e+01 \pm 3.66e+01$	+	$\mathbf{3.89e+01 \pm 1.31e+01}$	-
f_{13}	$1.27e+02 \pm 2.53e+01$	$1.50e+02 \pm 3.56e+01$	+	$1.11e+02 \pm 2.85e+01$	-	$\mathbf{9.03e+01 \pm 2.85e+01}$	-
f_{14}	$\mathbf{1.28e+02 \pm 3.57e+01}$	$1.20e+03 \pm 4.25e+02$	+	$3.06e+02 \pm 6.43e+01$	+	$1.47e+03 \pm 5.53e+02$	+
f_{15}	$\mathbf{3.54e+03 \pm 7.20e+02}$	$4.01e+03 \pm 7.00e+02$	+	$6.47e+03 \pm 3.23e+02$	+	$3.86e+03 \pm 6.71e+02$	+
f_{16}	$1.79e+00 \pm 5.24e-01$	$1.32e+00 \pm 8.61e-01$	-	$2.39e+00 \pm 2.61e-01$	+	$\mathbf{1.02e+00 \pm 4.40e-01}$	-
f_{17}	$\mathbf{3.25e+01 \pm 3.45e-01}$	$6.89e+01 \pm 1.24e+01$	+	$4.20e+01 \pm 1.50e+00$	+	$5.84e+01 \pm 8.44e+00$	+
f_{18}	$8.37e+01 \pm 1.71e+01$	$8.31e+01 \pm 1.66e+01$	=	$1.85e+02 \pm 1.12e+01$	+	$\mathbf{6.37e+01 \pm 1.08e+01}$	-
f_{19}	$\mathbf{2.14e+00 \pm 5.79e-01}$	$9.10e+00 \pm 4.94e+00$	+	$3.97e+00 \pm 4.09e-01$	+	$3.07e+00 \pm 7.81e-01$	+
f_{20}	$\mathbf{1.04e+01 \pm 7.06e-01}$	$1.09e+01 \pm 7.97e-01$	+	$1.20e+01 \pm 2.95e-01$	+	$1.12e+01 \pm 8.91e-01$	+
f_{21}	$3.22e+02 \pm 7.32e+01$	$3.09e+02 \pm 7.63e+01$	+	$3.00e+02 \pm 8.13e+01$	-	$\mathbf{2.96e+02 \pm 6.52e+01}$	-
f_{22}	$\mathbf{2.73e+02 \pm 8.63e+01}$	$1.11e+03 \pm 5.46e+02$	+	$1.12e+03 \pm 2.32e+02$	+	$1.85e+03 \pm 5.87e+02$	+
f_{23}	$\mathbf{3.93e+03 \pm 7.70e+02}$	$4.47e+03 \pm 7.32e+02$	+	$6.87e+03 \pm 3.26e+02$	+	$4.47e+03 \pm 6.85e+02$	+
f_{24}	$2.31e+02 \pm 8.13e+00$	$2.31e+02 \pm 1.11e+01$	=	$\mathbf{2.05e+02 \pm 7.41e+00}$	-	$2.43e+02 \pm 1.23e+01$	+
f_{25}	$2.73e+02 \pm 1.41e+01$	$2.75e+02 \pm 1.55e+01$	=	$\mathbf{2.64e+02 \pm 1.93e+01}$	-	$2.95e+02 \pm 8.81e+00$	+
f_{26}	$\mathbf{2.00e+02 \pm 1.59e-03}$	$2.16e+02 \pm 4.31e+01$	+	$2.00e+02 \pm 1.35e-02$	+	$2.02e+02 \pm 1.46e+01$	+
f_{27}	$6.82e+02 \pm 9.82e+01$	$6.55e+02 \pm 1.13e+02$	=	$\mathbf{5.41e+02 \pm 1.57e+02}$	-	$8.74e+02 \pm 9.41e+01$	+
f_{28}	$\mathbf{2.96e+02 \pm 2.80e+01}$	$3.11e+02 \pm 1.11e+02$	+	$3.00e+02 \pm 9.27e-13$	-	$3.00e+02 \pm 1.29e-12$	=

TABLE IV. AVERAGE ERROR ± STANDARD DEVIATION AND STATISTIC COMPARISON (REFERENCE: JESDE) FOR JESDE AGAINST MDE-PBX, JDE, AND SADE ON CEC2013 [29] IN 50 DIMENSIONS. THE BOLD FONT INDICATES THE ALGORITHM WITH THE LOWEST AVERAGE ERROR.

	jESDE	MDE-pBX		jDE		SADE	
f_1	$\mathbf{0.00e+00 \pm 0.00e+00}$	$3.34e-11 \pm 2.60e-10$	+	$0.00e+00 \pm 2.11e-13$	+	$2.27e-13 \pm 4.38e-13$	+
f_2	$\mathbf{2.77e+05 \pm 9.88e+04}$	$9.06e+05 \pm 4.90e+05$	+	$1.13e+06 \pm 3.48e+05$	+	$2.33e+06 \pm 9.86e+05$	+
f_3	$5.45e+07 \pm 6.17e+07$	$1.42e+08 \pm 1.57e+08$	+	$\mathbf{6.42e+06 \pm 1.26e+07}$	-	$9.91e+08 \pm 9.15e+08$	+
f_4	$\mathbf{1.27e+01 \pm 1.24e+01}$	$1.09e+03 \pm 8.33e+02$	+	$2.01e+03 \pm 8.70e+02$	+	$1.80e+04 \pm 4.64e+03$	+
f_5	$\mathbf{1.14e-13 \pm 1.14e-14}$	$2.54e-05 \pm 2.52e-04$	+	$1.14e-13 \pm 3.01e-14$	+	$1.14e-13 \pm 5.63e-13$	+
f_6	$4.73e+01 \pm 1.76e+01$	$5.67e+01 \pm 2.24e+01$	+	$\mathbf{4.35e+01 \pm 5.68e-01}$	=	$4.49e+01 \pm 5.34e+00$	+
f_7	$6.63e+01 \pm 1.22e+01$	$6.81e+01 \pm 1.22e+01$	=	$\mathbf{1.88e+01 \pm 9.88e+00}$	-	$7.95e+01 \pm 1.24e+01$	+
f_8	$2.11e+01 \pm 3.81e-02$	$2.12e+01 \pm 4.36e-02$	+	$\mathbf{2.11e+01 \pm 3.36e-02}$	-	$2.12e+01 \pm 8.76e-02$	+
f_9	$4.45e+01 \pm 5.49e+00$	$\mathbf{4.27e+01 \pm 6.99e+00}$	-	$6.08e+01 \pm 4.53e+00$	+	$5.65e+01 \pm 5.07e+00$	+
f_{10}	$\mathbf{9.63e-02 \pm 5.16e-02}$	$4.09e-01 \pm 5.57e-01$	+	$9.68e-02 \pm 4.03e-02$	=	$4.00e-01 \pm 9.28e-01$	+
f_{11}	$3.58e-01 \pm 1.89e+00$	$1.21e+02 \pm 2.97e+01$	+	$\mathbf{1.99e-02 \pm 1.39e-01}$	+	$6.00e+01 \pm 2.07e+01$	+
f_{12}	$1.96e+02 \pm 4.28e+01$	$1.62e+02 \pm 3.45e+01$	-	$1.52e+02 \pm 8.51e+01$	-	$\mathbf{9.17e+01 \pm 2.24e+01}$	-
f_{13}	$3.16e+02 \pm 4.81e+01$	$3.22e+02 \pm 5.39e+01$	=	$2.83e+02 \pm 4.39e+01$	-	$\mathbf{2.11e+02 \pm 3.86e+01}$	-
f_{14}	$\mathbf{2.42e+02 \pm 8.16e+01}$	$2.79e+03 \pm 8.06e+02$	+	$1.62e+03 \pm 2.20e+02$	+	$3.46e+03 \pm 7.81e+02$	+
f_{15}	$7.59e+03 \pm 1.39e+03$	$\mathbf{7.58e+03 \pm 8.01e+02}$	=	$1.27e+04 \pm 5.10e+02$	+	$7.75e+03 \pm 7.48e+02$	+
f_{16}	$2.31e+00 \pm 5.52e-01$	$1.93e+00 \pm 8.76e-01$	-	$3.20e+00 \pm 2.81e-01$	+	$\mathbf{1.70e+00 \pm 8.61e-01}$	-
f_{17}	$\mathbf{5.58e+01 \pm 6.19e-01}$	$1.79e+02 \pm 3.56e+01$	+	$9.31e+01 \pm 4.03e+00$	+	$1.10e+02 \pm 1.66e+01$	+
f_{18}	$2.07e+02 \pm 3.94e+01$	$1.86e+02 \pm 3.17e+01$	-	$3.73e+02 \pm 1.43e+01$	+	$\mathbf{1.13e+02 \pm 1.52e+01}$	-
f_{19}	$\mathbf{6.46e+00 \pm 2.03e+00}$	$3.94e+01 \pm 2.10e+01$	+	$8.95e+00 \pm 7.71e-01$	+	$7.91e+00 \pm 4.00e+00$	+
f_{20}	$\mathbf{1.93e+01 \pm 8.29e-01}$	$2.01e+01 \pm 9.17e-01$	+	$2.18e+01 \pm 3.20e-01$	+	$2.07e+01 \pm 1.09e+00$	+
f_{21}	$8.22e+02 \pm 3.61e+02$	$8.91e+02 \pm 3.44e+02$	+	$\mathbf{7.98e+02 \pm 4.26e+02}$	=	$8.54e+02 \pm 3.97e+02$	+
f_{22}	$\mathbf{3.94e+02 \pm 1.87e+02}$	$3.22e+03 \pm 1.06e+03$	+	$2.99e+03 \pm 4.23e+02$	+	$4.30e+03 \pm 8.32e+02$	+
f_{23}	$\mathbf{8.17e+03 \pm 1.20e+03}$	$9.08e+03 \pm 1.05e+03$	+	$1.34e+04 \pm 4.84e+02$	+	$9.04e+03 \pm 1.10e+03$	+
f_{24}	$2.88e+02 \pm 1.25e+01$	$2.88e+02 \pm 1.56e+01$	=	$\mathbf{2.38e+02 \pm 1.43e+01}$	-	$3.06e+02 \pm 1.88e+01$	+
f_{25}	$\mathbf{3.64e+02 \pm 1.19e+01}$	$3.68e+02 \pm 1.48e+01$	+	$3.68e+02 \pm 3.15e+01$	+	$3.92e+02 \pm 1.09e+01$	+
f_{26}	$2.73e+02 \pm 9.61e+01$	$3.55e+02 \pm 7.46e+01$	+	$\mathbf{2.00e+02 \pm 1.27e+00}$	+	$3.41e+02 \pm 1.10e+02$	+
f_{27}	$1.32e+03 \pm 1.20e+02$	$1.23e+03 \pm 1.49e+02$	-	$\mathbf{1.20e+03 \pm 3.97e+02}$	-	$1.62e+03 \pm 1.34e+02$	+
f_{28}	$5.36e+02 \pm 6.69e+02$	$5.05e+02 \pm 5.99e+02$	+	$\mathbf{4.00e+02 \pm 2.95e-13}$	-	$4.00e+02 \pm 1.25e-12$	+

TABLE V. AVERAGE ERROR ± STANDARD DEVIATION AND STATISTIC COMPARISON (REFERENCE: JESDE) FOR JESDE AGAINST MDE-PBX, JDE AND SADE ON CEC2010 [30] IN 1000 DIMENSIONS. THE BOLD FONT INDICATES THE ALGORITHM WITH THE LOWEST AVERAGE ERROR.

	jESDE	MDE-pBX		jDE		SADE	
f_1	$7.30e+07 \pm 1.10e+08$	$1.05e+09 \pm 6.58e+08$	+	$\mathbf{3.32e-04 \pm 3.15e-03}$	-	$2.89e+07 \pm 1.02e+08$	-
f_2	$7.05e+03 \pm 1.92e+02$	$7.02e+03 \pm 2.38e+02$	=	$\mathbf{2.36e+03 \pm 3.19e+02}$	-	$5.55e+03 \pm 2.99e+02$	-
f_3	$1.42e+01 \pm 4.35e-01$	$1.93e+01 \pm 4.76e-02$	+	$\mathbf{1.34e+01 \pm 7.00e-01}$	-	$1.89e+01 \pm 2.83e-01$	+
f_4	$1.19e+12 \pm 4.20e+11$	$3.21e+12 \pm 9.76e+11$	+	$\mathbf{9.30e+11 \pm 2.80e+11}$	-	$1.95e+12 \pm 8.82e+11$	+
f_5	$1.66e+08 \pm 2.85e+07$	$1.54e+08 \pm 2.77e+07$	-	$\mathbf{7.87e+07 \pm 1.45e+07}$	-	$1.03e+08 \pm 1.83e+07$	-
f_6	$2.99e+06 \pm 9.45e+05$	$3.65e+06 \pm 1.75e+06$	+	$\mathbf{1.42e+01 \pm 5.94e-01}$	-	$9.16e+05 \pm 1.21e+06$	-
f_7	$7.70e+05 \pm 1.39e+06$	$6.79e+06 \pm 1.01e+07$	+	$\mathbf{3.99e+01 \pm 2.31e+02}$	-	$1.01e+08 \pm 2.36e+08$	+
f_8	$6.83e+07 \pm 3.93e+07$	$2.03e+08 \pm 1.63e+08$	+	$\mathbf{4.97e+07 \pm 2.48e+07}$	-	$7.08e+07 \pm 3.71e+07$	=
f_9	$3.20e+08 \pm 2.47e+08$	$1.68e+09 \pm 1.00e+09$	+	$\mathbf{5.09e+07 \pm 4.67e+06}$	-	$2.11e+08 \pm 2.93e+08$	-
f_{10}	$7.51e+03 \pm 2.09e+02$	$7.33e+03 \pm 2.55e+02$	-	$\mathbf{4.46e+03 \pm 7.21e+02}$	-	$6.22e+03 \pm 3.15e+02$	-
f_{11}	$2.11e+02 \pm 1.42e+00$	$2.06e+02 \pm 2.40e+00$	-	$\mathbf{1.06e+02 \pm 1.51e+01}$	-	$2.05e+02 \pm 4.34e+00$	-
f_{12}	$\mathbf{1.16e+05 \pm 1.93e+04}$	$2.92e+05 \pm 6.60e+04$	+	$1.75e+06 \pm 2.27e+06$	+	$3.15e+05 \pm 1.36e+05$	+
f_{13}	$5.19e+07 \pm 2.07e+08$	$2.88e+09 \pm 3.17e+09$	+	$\mathbf{1.15e+03 \pm 2.00e+02}$	-	$5.67e+07 \pm 2.48e+08$	-
f_{14}	$4.96e+08 \pm 4.22e+07$	$1.04e+09 \pm 1.97e+08$	+	$\mathbf{1.69e+08 \pm 1.03e+07}$	-	$3.77e+08 \pm 1.13e+08$	-
f_{15}	$7.66e+03 \pm 2.07e+02$	$7.44e+03 \pm 2.80e+02$	-	$\mathbf{5.55e+03 \pm 2.39e+02}$	-	$6.49e+03 \pm 2.38e+02$	-
f_{16}	$3.89e+02 \pm 4.37e+00$	$3.84e+02 \pm 1.22e+00$	-	$\mathbf{3.28e+02 \pm 2.06e+01}$	-	$3.82e+02 \pm 2.00e+00$	-
f_{17}	$\mathbf{2.40e+05 \pm 2.46e+04}$	$4.35e+05 \pm 8.33e+04$	+	$3.61e+06 \pm 3.71e+06$	=	$6.37e+05 \pm 2.00e+05$	+
f_{18}	$2.76e+09 \pm 3.35e+09$	$3.73e+10 \pm 1.95e+10$	+	$\mathbf{2.70e+03 \pm 5.36e+02}$	-	$7.60e+08 \pm 1.14e+09$	-
f_{19}	$\mathbf{8.96e+05 \pm 6.56e+04}$	$9.22e+05 \pm 1.06e+05$	+	$1.89e+07 \pm 4.28e+06$	+	$2.11e+06 \pm 1.61e+05$	+
f_{20}	$2.07e+09 \pm 2.44e+09$	$4.18e+10 \pm 2.02e+10$	+	$\mathbf{2.31e+03 \pm 1.74e+02}$	-	$2.26e+09 \pm 3.42e+09$	-

[18] R. Mallipeddi, G. Iacca, P. N. Suganthan, F. Neri, and E. Mininno, "Ensemble Strategies in Compact Differential Evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2011, pp. 1972–1977.

[19] R. Mallipeddi and P. Suganthan, "Differential evolution algorithm with ensemble of populations for global numerical optimization," *OPSEARCH*, vol. 46, no. 2, pp. 184–213, 2009.

[20] R. Mallipeddi and P. Suganthan, "Ensemble of constraint handling techniques," *Evolutionary Computation, IEEE Transactions on*, vol. 14, no. 4, pp. 561–579, 2010.

[21] G. Iacca, F. Neri, E. Mininno, Y. S. Ong, and M. H. Lim, "Ockham's Razor in Memetic Computing: Three Stage Optimal Memetic Exploration," *Information Sciences*, vol. 188, pp. 17–43, 2012.

[22] J. Derrac, S. Garcia, S. Hui, F. Herrera, and P. Suganthan, "Statistical analysis of convergence performance throughout the evolutionary search: A case study with SaDE-MMTS and Sa-EPSDE-MMTS," in *Differential Evolution (SDE), 2013 IEEE Symposium on*, April 2013, pp. 151–156.

[23] L.-Y. Tseng and C. Chen, "Multiple trajectory search for Large Scale Global Optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2008, pp. 3052–3059.

[24] K. Price, "An Introduction to Differential Evolution," in *New Ideas in Optimization*, D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli, and K. V. Price, Eds. McGraw-Hill, 1999, pp. 79–108.

[25] N. Krasnogor, "Toward Robust Memetic Algorithms," in *Recent Advances in Memetic Algorithms*, ser. Studies in Fuzzines and Soft Computing, W. E. Hart, N. Krasnogor, and J. E. Smith, Eds. Berlin, Germany: Springer, 2004, pp. 185–207.

[26] F. Neri and C. Cotta, "Memetic algorithms and memetic computing optimization: A literature review," *Swarm and Evolutionary Computation*, vol. 2, pp. 1–14, 2012.

[27] M. Weber, F. Neri, and V. Tirronen, "A Study on Scale Factor in Distributed Differential Evolution," *Information Sciences*, vol. 181, no. 12, pp. 2488–2511, 2011.

[28] S. Islam, S. Das, S. Ghosh, S. Roy, and P. Suganthan, "An Adaptive Differential Evolution Algorithm With Novel Mutation and Crossover Strategies for Global Numerical Optimization," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 42, no. 2, pp. 482–500, 2012.

[29] J. J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernndez-Daz, "Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization," Zhengzhou University and Nanyang Technological University, Zhengzhou China and Singapore, Tech. Rep. 201212, 2013.

[30] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization," University of Science and Technology of China, Tech. Rep., 2010.

[31] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[32] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2008.