2013 IEEE Congress on Evolutionary Computation
June 20-23, Cancún, México

Investigation of Self-adaptive Differential Evolution on the CEC-2013 Real-Parameter Single-Objective Optimization Testbed

A. K. Qin*†, Xiaodong Li*, Hong Pan†, Siyu Xia†
*School of Computer Science and Information Technology, RMIT University, Melbourne, 3001, Victoria, Australia
Email:{kai.qin, xiaodong.li}@rmit.edu.au, mspanhong@hotmail.com, xia081@gmail.com
†School of Automation, Southeast University, Nanjing, China, 210096

***Abstract*—Self-adaptive differential evolution (SaDE) is a well-known DE variant, which has received considerable attention since it was developed. SaDE gradually adapts its trial vector generation strategy and the accompanying parameter setting via learning the preceding performance of multiple candidate strategies and their associated parameter settings. This work systematically investigates SaDE on the CEC-2013 real-parameter single-objective optimization testbed. Parameter sensitivity analysis is carried out by using advanced statistical hypothesis testing methods, aiming to detect statistically significantly superior parameter settings. This analysis reveals that SaDE is actually less sensitive to the parameter choice since quite a number of parameter settings can lead to the statistically significantly better performance than the other settings. Based on this finding, we report SaDE's performance using one of the parameter settings advocated by sensitivity analysis and statistically compare this performance with that of a widely used classic DE (DE/rand/1/bin). The comparison results significantly favor SaDE.**

I. Introduction

Differential evolution (DE) [1]–[3], as proposed by Storn and Price in 1995, is an effective and powerful optimization method for solving real-parameter black-box optimization problems. Over the past decades, there has been a large body of research works on improving the performance of DE [4]–[13], empirically or theoretically studying the behavior of DE [14], applying various DE based algorithms to challenging optimization tasks [3]. It is well-known that DE's performance depends on the choice of search strategies and the accompanying control parameters, which is highly problem-dependent. Furthermore, a search strategy even equipped with the best-calibrated parameter setting may not demonstrate consistent effectiveness in various sub-regions of the search space explored at different optimization stages. This has motivated many research efforts on investigating the adaption of search strategies and parameters in DE.

Among these works, a self-adaptive differential evolution algorithm, proposed in [9], [10], deserves special mention. This algorithm is capable of gradually adapting the its trial vector generation strategy and the accompanying parameter setting via learning the preceding performance of multiple candidate strategies and their associated parameter settings. SaDE involves only two control parameters, i.e., population size (NP) and learning period (LP). Since its invention, SaDE has received considerable attention as evidenced by its high citation counts (the most cited paper among all papers published in the 2009 *IEEE Transactions on Evolutionary Computation* (TEVC)) and the winner of the 2012 IEEE TEVC outstanding paper award. Many derivatives of the original SaDE algorithm have been proposed with an aim to further improve SaDE's performance [11], [13], [15].

This paper systematically investigates SaDE's performance on the newly proposed CEC-2013 real-parameter single-objective optimization testbed. This testbed consists of 28 test functions, improving its predecessor CEC-2005 with additional test functions, non-linear transformations and modified definitions of composition functions. In this paper, we evaluate SaDE, using a range of 30 potentially effective parameter settings, on all 28 CEC-2013 test functions at three dimension sizes (10D, 30D and 50D). We also conduct parameter sensitivity analysis over all 28 functions at each dimension size using advanced statistical hypothesis testing methods (the Iman and Davenport test followed by the Hochberg procedure [16], [17]). The results reveal that SaDE is less sensitive to the parameter choice, as evidenced by quite a number of parameter settings ([NP, LP] $\in$ [50, 70, 100] $\times$ [30, 50, 70]) leading to the statistically significantly similar performance which is significantly better than the other settings. This finding is consistent with the conclusions in the original work [10] but is deduced in a statistically rigorous manner. Furthermore, we report the results using one of the superior parameter settings (NP:50, LP:50) advocated by sensitivity analysis, and compare these results with those of a widely-used classic DE (DE/rand/1/bin) employing a typically suggested parameter setting (NP:50, CR:0.9, F:0.5) using the Wilcoxon's signed rank test [18].

The remaining paper is organized as the following. First, a brief review on DE is provided in Section II, followed by a description of SaDE in III. Section IV reports and analyzes experimental results. Section V concludes this paper.

II. Differential Evolution

As a highly effective algorithm for solving real-parameter black-box optimization problems, DE together with its variants has been thoroughly described in [1]–[3]. Let us consider a real-parameter single-objective minimization problem: for a function $f : X \subseteq \mathcal{R}^D \rightarrow Y \subseteq \mathcal{R}$, the aim is to find $\mathbf{x}^* \in X$ such that $\forall \mathbf{x} \in X, f(\mathbf{x}^*) \leq f(\mathbf{x})$. Suppose $\mathbf{x}_{i,g} = \{x^1_{i,g}, \ldots, x^D_{i,g}\}$ represent the i-th decision vector of dimension size D at the g-th generation. Let $\mathbf{P}^g =$

Algorithm 1 Trial Vector Generation Procedure in DE/rand/1/bin

Input: *NP*, *CR*, *F*, $\mathbf{x}_{i,g} = \{x_{i,g}^1, \ldots, x_{i,g}^D\}$
Output: $\mathbf{u}_{i,g} = \{u_{i,g}^1, \ldots, u_{i,g}^D\}$

Randomly select in $\{1, \ldots, NP\}$ three mutually exclusive indices that are distinct from i:
  do $r_1 = ceil(rand_u(1, NP))$
  while $r_1 \neq i$
  do $r_2 = ceil(rand_u(1, NP))$
  while $r_2 \neq i$ and $r_2 \neq r_1$
  do $r_3 = ceil(rand_u(1, NP))$
  while $r_3 \neq i$ and $r_3 \neq r_1$ and $r_3 \neq r_2$
$j_{rand} = ceil(rand_u(1, D))$
for $j = 1 \rightarrow D$ **do**

$$u_{i,g}^j = \begin{cases} x_{r_1,g}^j + F \cdot (x_{r_2,g}^j - x_{r_3,g}^j) & \text{if } rand_u(0,1) \leq CR \text{ or } j = j_{rand} \\ x_{i,g}^j & \text{otherwise} \end{cases}$$

end for

NOTE: (1) $rand_u(a, b)$ is a uniform random number generator sampling in $[a, b]$.
(2) $ceil(c)$ takes on the smallest integer larger than or equal to c.

$\{\mathbf{x}_{1,g}, \ldots, \mathbf{x}_{NP,g}\}$ denote the population of size *NP* at the g-th generation, and its offspring population represented by $\mathbf{P}^{g+1} = \{\mathbf{x}_{1,g+1}, \ldots, \mathbf{x}_{NP,g+1}\}$. In the initial population of DE, i.e., $\mathbf{P}^0$, each decision vector $\mathbf{x}_{i,0}, i = 1, \ldots, NP$ is generated by sampling along each element of the decision vector a uniform random number between the lower and upper bounds of the corresponding vector element. At each generation, e.g., the g-th generation, an offspring $\mathbf{u}_{i,g}$ (so-called trial vector) is generated with respect to each decision vector $\mathbf{x}_{i,g}$ (so-called target vector) in the population for the current generation using mutation and recombination operations. The pseudo-code of this trial vector generation procedure in the context of a classic DE strategy (DE/rand/1/bin) is depicted in Algorithm 1. Then, DE employs a greedy replacement scheme to generate the population at the (g+1)-th generation, i.e., $\mathbf{P}^{g+1}$. Specifically, the "offspring" trial vector $\mathbf{u}_{i,g}$ will enter $\mathbf{P}^{g+1}$ if $f(\mathbf{u}_{i,g})$ is smaller or equal to $f(\mathbf{x}_{i,g})$. Otherwise, the "parent" target vector $\mathbf{x}_{i,g}$ will be retained in $\mathbf{P}^{g+1}$. The above procedure is repeated generation by generation until certain termination criteria are met. For more detailed information on DE's implementations, readers can refer to [1], [2].

DE differs from other evolutionary algorithms by its unique self-referential mutation scheme. Classic DE algorithms are typically denoted by "DE/x/y/z" where "x" defines the base vector generation scheme, "y" defines how many pairs of population members are used to establish the vector difference and "z" defines the recombination scheme. Among classic DE strategies, DE/rand/1/bin is most widely used, which involves three control parameters: population size (NP), crossover rate (CR) and mutation scale factor (F). Previous works have empirically or theoretically demonstrated that the performance of classic DE algorithms crucially depends on the proper parameter setting, which is highly problem-dependent. Although the trial-and-error scheme may deduce certain satisfying parameter setting, it may require demanding computational costs, which is not practically feasible. Existing research works on DE's parameter configuration can be generally divided into three categories: (1) Fixed schemes use a fixed parameter setting derived from theoretical or empirical studies during the searching process [1], [2], [14]; (2) *Control schemes* use certain pre-specified rules to update the parameter setting during the searching process [6]; (3) *Adaptive schemes* adaptively alter the parameter setting by learning its historical impact on the searching performance during the searching process [4], [5], [7]–[11], [13], which have become recent research trends.

III. Self-adaptive Differential Evolution (SaDE)

In addition to the parameter setting, DE's performance also relies on the choice of trial vector generation strategies, which is highly problem-dependent. For example, DE/rand/1/bin is more proficient in solving multi-modal problems than DE/best/1/bin while the latter can efficiently solve uni-modal problems. Furthermore, a trial vector generation strategy can demonstrate its intrinsic efficacy only if it is equipped with suitable parameter settings. As a result, the trial-and-error scheme is still infeasible due to prohibitive computational costs. On the other hand, one single strategy even armed with well-calibrated parameter settings cannot guarantee consistent effectiveness at different searching stages since sub-regions of the search space explored at varying searching stages may not always prefer this strategy. This fact motivated the study of strategy adaptation in DE.

A differential evolution with strategy adaptation algorithm, so-called SaDE, was proposed in [9], [10], which can gradually adapt the employed trial vector generation strategy and the accompanying parameter setting in favor of varying searching stages via learning the preceding performance of multiple candidate strategies and their associated parameter settings. Specifically, SaDE features a pool of potentially effective yet complementary trial vector generation strategies. During the population's evolution, with respect to each target vector in the population for the current generation, one strategy will be chosen from this pool based on the selection probabilities of all available strategies, which are calculated from the success rate of each strategy for generating promising trial vectors (those that can enter the population for the next generation) within a certain number of preceding generations, so-called learning period (LP). This selected strategy is then applied to the corresponding target vector to generate the trial vector. As for the parameter setting associated with such a selected strategy, we only consider adapting CR while randomizing F

and manually pre-specifying NP. The rationale behind this is as follows:

1) CR is sensitive to the the properties of the searching landscape and thus should better be adapted for varying searching stages. SaDE archives the CR values associated with each strategy which had generated promising trial vectors within the preceding LP generations. The median of those recorded CR values with respect to each strategy is calculated at the end of the current generation, and used as the mean value in the normal distribution with standard deviation 0.1 to generate the CR values used by the corresponding strategy in the next generation.
2) F is closely related to the convergence speed. Usually, large F values favor exploration and thus slow down the convergence while small values favor exploitation and thus speed up the convergence. SaDE randomizes F according to the normal distribution with mean value 0.5 and standard deviation 0.3, which makes the values of F to almost fall into [-0.4, 1.4]. As a result, both exploration (large F values) and exploitation (small F values) capabilities are well retained throughout the searching course.
3) NP is influenced by the problem scale and complexity as well as the available computational budget. SaDE leaves it as a user-specified parameter, which should be determined as per any knowledge about the problem (e.g., the problem dimension size) and the computational budget.

SaDE allows the initial LP generations to accumulate the searching performance to be learnt. During this period, all strategy selection probabilities are set to be equal and the mean value in the normal distribution for generating the CR values is set to 0.5. To avoid the invalid calculation of selection probabilities when the success rates of all strategies are null or when any strategy is never selected within the preceding LP generations, a small constant value (0.01) is introduced as depicted in Algorithm 2, which illustrates the pseudo-code of the original SaDE algorithm.

IV. Experiments

We test SaDE under six NP values (30, 50, 70, 100, 200 and 300) and five LP values (5, 10, 30, 50, 70) on 28 test functions contained in the CEC-2013 testbed at three problem dimension sizes (10D, 30D and 50D). We report all performance measures required in the protocol of the CEC-2013 testbed [19] as well as the success rate, the expected running time to succeed (ERT) [20], [21] and the empirical cumulative distribution function (ECDF) [21] of the number of function evaluations at success and the best achieved object function error value at termination.

To find out a statistically reliable rule-of-thumb on how to choose the parameters of SaDE, we conduct parameter sensitivity analysis using advanced statistical hypothesis testing methods, i.e., the Iman and Davenport test followed by the Hochberg procedure [16], [17], to compare 30 parameter settings over all 28 test functions at 10D, 30D and 50D, respectively. We report SaDE's performance in regard to one of the superior parameter settings identified by such sensitivity analysis. Furthermore, we compare the performance of SaDE using this identified parameter setting with that of DE/rand/1/bin using a typically used parameter setting (NP:50, CR:0.9, F:0.5) on all 28 test functions at 10D, 30D and 50D, respectively, using the Wilcoxon's signed rank test [18].

A. Experimental Setup

The CEC-2013 testbed involves 28 numerical test functions, categorized into uni-modal functions (f_1-f_5), multi-modal functions (f_6-f_{20}) and composition functions (f_{21}-f_{28}), which extends its predecessor CEC-2005 testbed. The detailed description of the CEC-2013 testbed can be referred to in [19].

We test SaDE under 30 parameter settings ([NP, LP] $\in$ [30, 50, 70, 100, 200, 300] $\times$ [5, 10, 30, 50, 70]) on each of 28 test functions at three problem dimension sizes (10D, 30D and 50D), respectively. These settings include those of typical parameter configurations for SaDE [10]. We also test DE/rand/1/bin using a commonly suggested parameter setting (NP:50, CR:0.9, F:0.5) for the purpose of the performance comparison with SaDE.

SaDE using 30 parameter settings and DE/rand/1/bin using the parameter setting (NP:50, CR:0.9, F:0.5) are executed 51 times with respect to each test function at each problem dimension size. Each of 51 runs uses distinct random seeds. For any individual run, SaDE using each of 30 parameter settings as well as DE/rand/1/bin using the parameter setting (NP:50, CR:0.9, F:0.5) employ the same random seed.

Two stopping criteria are used here [19]: (1) the maximum number of function evaluations (maxFEvals), set to 10^4 times the problem dimension size, is reached. (2) The object function error value (FEV), defined as the difference between the objective function value of the best solution found so far and that of the globally optimal solution, is less than or equal to 10^{-8}. In this case, **we set FEV to** 10^{-8} instead of zero as suggested in the CEC-2013 protocol since the latter way may dramatically decrease the average FEV at termination if only a few runs reach 10^{-8}.

We use MATLAB to implement all algorithms. The algorithm execution platform is a Windows PC with the Intel Xeon E5-2630 CPU at 2.3 GHz.

The algorithm's performance is measured by the following:

1) the best, worse, median and mean (standard deviation) of the FEVs achieved when the algorithm terminates over 51 runs;
2) the success rate (SR) over 51 runs. An execution run is claimed to succeed once the algorithm achieves the FEV smaller than 10^{-8};
3) the expected running time to succeed (ERT) [20], [21]. This performance index estimates the expected number of function evaluations to succeed. It is computed by the total number of function evaluations when the algorithm succeeds or terminates (without succeeding) summed over 51 runs and divided by the total number of successful runs. If all runs fail, this measure is invalid;
4) the empirical cumulative distribution function (ECDF) [21] of the number of executed function evaluations at success and the best achieved FEVs at termination over 51 runs of all 28 test functions at three problem dimension sizes (10D, 30D and 50D), respectively. This measure illustrates the efficiency and effectiveness of an

TABLE I. PARAMETER SENSITIVITY ANALYSIS RESULTS BY USING ADVANCED STATISTICAL HYPOTHESIS TESTING METHODS ON ALL 28 CEC-2013 TEST FUNCTIONS AT PROBLEM DIMENSION SIZES 10D, 30D AND 50D, RESPECTIVELY. AMONG 30 TESTED PARAMETER SETTINGS ([NP, LP] $\in$ [30, 50, 70, 200, 300] $\times$ [5, 10, 30, 50, 70]), THOSE LEADING TO THE STATISTICALLY BETTER PERFORMANCE (AT THE SIGNIFICANCE LEVEL 0.05) OVER OTHERS WITH RESPECT TO 10D, 30D AND 50D PROBLEMS ARE DENOTED BY 10, 30 AND 50 AND SEPARATED IN ORDER BY "/" IN THEIR CORRESPONDING TABLE CELLS. "-" INDICATES THE CORRESPONDING PARAMETER SETTING IS STATISTICALLY SIGNIFICANTLY WORSE THAN SOME OTHER PARAMETER SETTINGS FOR SOLVING ALL 28 TEST FUNCTIONS AT CERTAIN PROBLEM DIMENSION SIZES.

Parameter	**Population Size (NP)**					
Learning Period (LP)	**30**	**50**	**70**	**100**	**200**	**300**
5	-/-/-	-/-/-	-/-/-	-/-/-	-/-/-	-/-/-
10	-/-/-	-/-/-	10/-/-	-/-/50	-/-/-	-/-/-
30	-/-/-	10/30/50	10/30/50	10/30/50	-/-/50	-/-/-
50	-/-/-	10/30/50	10/30/50	10/30/50	-/-/50	-/-/-
70	-/-/-	10/30/50	10/30/50	-/30/50	-/-/50	-/-/-

TABLE II. PERFORMANCE (PFM) OF SADE USING ONE PARAMETER SETTING (NP:50 AND LP:50) ADVOCATED BY PARAMETER SENSITIVITY ANALYSIS ON 28 CEC-2013 TEST FUNCTIONS AT PROBLEM DIMENSION SIZES (DIM) 10D, 30D AND 50D, RESPECTIVELY. **BEST**, **WORST**, **MEDIAN**, **MEAN** (**STD**) REPRESENT THE BEST, WORST, MEDIAN, MEAN (STANDARD DEVIATION) OF THE FEVS AT EXECUTION TERMINATION OVER 51 RUNS, RESPECTIVELY. **SR** AND **ERT** STAND FOR THE SUCCESS RATE AND THE EXPECTED RUNNING TIME TO SUCCEED. **ERT** IS DENOTED BY "-" (INVALID) WHEN ALL 51 RUNS FAIL.

DIM	**PFM**	**f1**	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**	**f9**	**f10**	**f11**	**f12**	**f13**	**f14**
10D	**Best**	1.00e-08	1.00e-08	1.00e-08	1.00e-08	1.00e-08	1.00e-08	1.44e-06	2.02e+01	7.56e-04	1.00e-08	1.00e-08	9.95e-01	9.95e-01	1.00e-08
	Worst	1.00e-08	6.31e-02	5.48e+02	3.99e-03	1.00e-08	9.81e+00	4.18e+00	2.05e+01	5.25e+00	5.67e-02	1.00e-08	9.95e+00	1.25e+01	1.25e-01
	Median	1.00e-08	1.00e-08	4.23e-03	3.91e-07	1.00e-08	9.81e+00	7.06e-03	2.04e+01	1.11e+00	2.46e-02	1.00e-08	4.16e+00	4.87e+00	1.00e-08
	Mean	1.00e-08	1.66e-03	1.24e+01	1.83e-04	1.00e-08	5.77e+00	1.30e-01	2.04e+01	1.40e+00	2.28e-02	1.00e-08	4.48e+00	5.47e+00	1.59e-02
	Std	0.00e-00	9.12e-03	7.65e+01	6.37e-04	0.00e-00	4.88e+00	5.91e-01	6.65e-02	1.25e+00	1.61e-02	0.00e-00	1.57e+00	3.06e+00	3.02e-02
	SR	1.00	0.71	0.35	0.35	1.00	0.41	0.00	0.00	0.00	0.10	1.00	0.00	0.00	0.76
	ERT	1.08e+04	1.17e+05	2.36e+05	2.67e+05	1.43e+04	1.83e+05	-	-	-	9.57e+05	2.61e+04	-	-	7.80e+04
		f15	**f16**	**f17**	**f18**	**f19**	**f20**	**f21**	**f22**	**f23**	**f24**	**f25**	**f26**	**f27**	**f28**
	Best	1.74e+02	6.77e-01	1.01e+01	1.31e+01	2.15e-01	1.44e+00	2.00e+02	1.00e-08	5.35e+01	1.06e+02	1.08e+02	1.01e+02	3.00e+02	1.00e+02
	Worst	9.85e+02	1.50e+00	1.01e+01	3.23e+01	4.89e-01	3.02e+00	4.00e+02	3.50e+01	1.24e+03	2.09e+02	2.06e+02	2.00e+02	3.00e+02	3.00e+02
	Median	7.41e+02	1.11e+00	1.01e+01	2.28e+01	3.94e-01	2.20e+00	4.00e+02	8.84e+00	6.59e+02	2.00e+02	2.00e+02	1.06e+02	3.00e+02	3.00e+02
	Mean	6.86e+02	1.12e+00	1.01e+01	2.28e+01	3.76e-01	2.23e+00	3.96e+02	1.13e+01	6.55e+02	1.94e+02	1.98e+02	1.27e+02	3.00e+02	2.96e+02
	Std	1.99e+02	1.93e-01	2.76e-03	3.26e+00	5.65e-02	3.53e-01	2.80e+01	8.67e+00	2.36e+02	2.44e+01	1.29e+01	3.93e+01	5.45e-02	2.80e+01
	SR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00
	ERT	-	-	-	-	-	-	-	5.08e+06	-	-	-	-	-	-

DIM	**PFM**	**f1**	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**	**f9**	**f10**	**f11**	**f12**	**f13**	**f14**
30D	**Best**	1.00e-08	1.02e+04	2.39e+00	5.57e+00	1.00e-08	6.37e-03	3.24e+00	2.08e+01	1.19e+01	2.22e-02	1.00e-08	1.99e+01	3.59e+01	1.00e-08
	Worst	1.00e-08	8.85e+04	2.55e+07	7.97e+02	1.00e-08	7.23e+01	4.95e+01	2.10e+01	2.87e+01	4.73e-01	9.95e-01	6.24e+01	1.17e+02	1.18e+00
	Median	1.00e-08	3.30e+04	1.02e+06	4.75e+01	1.00e-08	8.74e+00	1.69e+01	2.09e+01	1.59e+01	1.16e-01	1.00e-08	3.18e+01	7.35e+01	4.16e-02
	Mean	1.00e-08	3.40e+04	3.32e+06	1.03e+02	1.00e-08	8.72e+00	1.92e+01	2.09e+01	1.69e+01	1.52e-01	5.85e-02	3.34e+01	7.21e+01	8.34e-02
	Std	0.00e-00	1.63e+04	5.33e+06	1.52e+02	0.00e-00	1.03e+01	1.06e+01	5.21e-02	3.81e+00	1.02e-01	2.36e-01	8.92e+00	2.02e+01	2.25e-01
	SR	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.94	0.00	0.00	0.12
	ERT	2.52e+04	-	-	-	4.15e+04	-	-	-	-	-	8.84e+04	-	-	2.46e+06
		f15	**f16**	**f17**	**f18**	**f19**	**f20**	**f21**	**f22**	**f23**	**f24**	**f25**	**f26**	**f27**	**f28**
	Best	4.09e+03	1.52e+00	3.04e+01	9.17e+01	1.28e+00	9.49e+00	2.00e+02	2.41e+01	2.92e+03	2.11e+02	2.48e+02	2.00e+02	4.22e+02	3.00e+02
	Worst	5.95e+03	3.02e+00	3.05e+01	1.42e+02	2.75e+00	1.20e+01	4.44e+02	2.45e+02	5.77e+03	2.47e+02	2.73e+02	3.32e+02	7.30e+02	3.00e+02
	Median	4.80e+03	2.39e+00	3.04e+01	1.25e+02	2.11e+00	1.10e+01	3.00e+02	1.12e+02	4.83e+03	2.25e+02	2.62e+02	2.00e+02	5.61e+02	3.00e+02
	Mean	4.82e+03	2.38e+00	3.04e+01	1.25e+02	2.08e+00	1.09e+01	3.04e+02	1.14e+02	4.82e+03	2.26e+02	2.62e+02	2.10e+02	5.66e+02	3.00e+02
	Std	4.08e+02	3.05e-01	2.82e-03	9.80e+00	2.51e-01	5.18e-01	7.68e+01	3.60e+01	4.72e+02	8.52e+00	5.19e+00	3.45e+01	7.24e+01	0.00e-00
	SR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	ERT	-	-	-	-	-	-	-	-	-	-	-	-	-	-

DIM	**PFM**	**f1**	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**	**f9**	**f10**	**f11**	**f12**	**f13**	**f14**
50D	**Best**	1.00e-08	7.34e+04	3.96e+05	6.51e+01	1.00e-08	1.82e+01	2.56e+01	2.10e+01	2.77e+01	4.68e-02	1.00e-08	7.26e+01	1.17e+02	3.75e-02
	Worst	1.00e-08	4.22e+05	1.62e+08	2.79e+03	1.00e-08	9.05e+01	7.96e+01	2.12e+01	5.66e+01	4.97e-01	1.29e+01	1.58e+02	2.85e+02	5.05e+00
	Median	1.00e-08	1.94e+05	1.79e+07	2.02e+02	1.00e-08	4.34e+01	4.88e+01	2.11e+01	3.58e+01	1.33e-01	1.00e-08	1.00e+02	2.02e+02	8.74e-02
	Mean	1.00e-08	1.93e+05	3.33e+07	3.45e+02	1.00e-08	4.51e+01	4.84e+01	2.11e+01	3.63e+01	1.62e-01	5.85e-01	1.03e+02	2.04e+02	4.34e-01
	Std	0.00e-00	6.94e+04	3.85e+07	4.63e+02	0.00e-00	9.31e+00	1.04e+01	3.65e-02	5.40e+00	9.17e-02	1.85e+00	1.85e+01	3.57e+01	8.44e-01
	SR	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.69	0.00	0.00	0.00
	ERT	4.30e+04	-	-	-	7.70e+04	-	-	-	-	-	3.49e+05	-	-	-
		f15	**f16**	**f17**	**f18**	**f19**	**f20**	**f21**	**f22**	**f23**	**f24**	**f25**	**f26**	**f27**	**f28**
	Best	8.48e+03	2.48e+00	5.08e+01	2.24e+02	3.71e+00	1.83e+01	2.00e+02	1.21e+01	8.74e+03	2.48e+02	3.02e+02	2.00e+02	9.28e+02	4.00e+02
	Worst	1.15e+04	3.68e+00	5.09e+01	3.10e+02	7.46e+00	2.17e+01	1.12e+03	5.64e+02	1.21e+04	3.05e+02	3.55e+02	3.99e+02	1.37e+03	3.58e+03
	Median	1.03e+04	3.19e+00	5.08e+01	2.60e+02	5.76e+00	2.05e+01	8.36e+02	1.81e+01	1.04e+04	2.73e+02	3.22e+02	2.00e+02	1.15e+03	4.00e+02
	Mean	1.02e+04	3.20e+00	5.08e+01	2.64e+02	5.78e+00	2.04e+01	7.74e+02	7.72e+01	1.03e+04	2.76e+02	3.23e+02	2.83e+02	1.16e+03	5.23e+02
	Std	6.93e+02	2.95e-01	1.05e-02	1.82e+01	9.42e-01	6.98e-01	3.77e+02	1.18e+02	8.44e+02	1.23e+01	1.08e+01	8.98e+01	1.01e+02	6.14e+02
	SR	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	ERT	-	-	-	-	-	-	-	-	-	-	-	-	-	-

optimization algorithm as computation budgets and target accuracies vary;

5) the computational complexity measured by CPU-seconds at three problem dimension sizes (10D, 30D and 50D), respectively [19].

B. Result Analysis

The original work on SaDE [10] included the empirical study on SaDE's parameter sensitivity, which, however, was not conducted in a statistically rigorous manner. In this paper, we make up this shortcoming by using advanced statistical

TABLE III. WILCOXON'S SIGNED RANK TEST RESULTS FOR COMPARING SADE (NP:50, LP:50) TO DE/RAND/1/BIN (NP:50, CR:0.9, F:0.5) AT THREE PROBLEM DIMENSION SIZES 10D, 30D AND 50D, RESPECTIVELY. $R-$ DENOTES THE SUM OF RANKS FOR THE TEST PROBLEMS IN WHICH SADE PERFORMS BETTER THAN DE/RAND/1/BIN, AND $R+$ REPRESENTS THE SUM OF RANKS FOR THE TEST PROBLEMS IN WHICH SADE PERFORMS WORSE THAN DE/RAND/1/BIN. LARGER RANKS INDICATE LARGER PERFORMANCE DISCREPANCY.

DIM	$R-$	$R+$	p_{value}
10D	5.641e+5	1.494e+5	0.000e+5
30D	6.724e+5	2.140e+5	0.000e+5
50D	6.374e+5	2.437e+5	0.000e+5

TABLE IV. COMPUTATIONAL COMPLEXITY MEASURED BY CPU SECONDS ([19]) AT 10D, 30D AND 50D, RESPECTIVELY. T_0 MEASURES THE COMPUTATION TIME OF BASIC OPERATIONS. T_1 MEASURES THE COMPUTATION TIME OF ONE EXECUTION RUN ON TEST FUNCTION f_{14}. $\widehat{T}_2$ CONSIDERS THE VARIATION OF T_1 IN DIFFERENT EXECUTION RUNS.

DIM	T_0	T_1	$\widehat{T}_2$	$(\widehat{T}_2 - T_1)/T_0$
$D = 10$	0.152	13.686	13.088	-3.932
$D = 30$		14.105	14.092	-0.083
$D = 50$		15.626	15.508	-0.776

hypothesis testing methods to carry out parameter sensitivity study in order to identify parameter settings (out of many possibilities) that can lead to the statistically significantly superior performance. More specifically, we first employ the Iman and Davenport test [16], [17] to compare 30 parameter settings over all 28 test functions to see whether there exist at least two settings that can produce statistically significantly different results. If this is the case, we then apply the Hochberg post-hoc procedure [16], [17] to further identify the exact statistically significantly superior settings. In our work, the significance level is set to 0.05.

Friedman's test is a well-known non-parametric two-way analysis of variance method [18], which can be used to detect among multiple algorithms whether there exists the statistically significant difference between the performance of at least two algorithms. It first converts the original performance measures of all algorithms in comparison to ranking values based on which the test statistic is computed. The calculated test statistic is then converted to the p-value according to the F distribution. By comparing this p-value to a pre-specified significant level, we can draw the conclusion on whether at least two compared algorithms demonstrate the statistically significantly different performance. The Iman and Davenport test [16], [17] modifies the Friedman's test statistic to reduce the undesirable conservative effect of the Friedman's test. Although the Iman and Davenport test as well as the Friedman's test is able to identify the existence of the statistically significant performance distinction among multiple compared algorithms, it cannot separate out which algorithms are significantly different from the others. To address this issue, some post-hoc procedures can be employed to perform pairwise performance comparisons under the control of the family-wise error rate (FWER) [16], [17]. In this work, the Hochberg procedure is used, as recommended in [16], [17]. To apply the Hochberg procedure, we need to first determine a control method, which is chosen as the algorithm with the lowest ranking value obtained from the Iman and Davenport test. For more detailed information on the Iman and Davenport test and the Hochberg post-hoc procedure, readers can refer to [16]–[18].

As mentioned in [17], the compared algorithm number and the test problem number would impact the reliability and power of the tests described above. As advocated in [17], the tested problem number should be not smaller than two times and not larger than eight times the compared algorithm number. For population-based stochastic algorithms that we are investigating, distinct random seeds may make the population to traverse in different parts of the search space and thus lead to the distinct performance even for solving the same optimization problem. This insight allows us to treat applying different random seeds to the same test function as different test problems. Following this understanding, for each test function at any tested problem dimension size, we choose 6 execution runs under 6 distinct random seeds to serve as 6 different test problems, which produces a total of 168 test problems for each tested problem dimension size. This makes the test problem number 5.6 times the tested parameter setting number, which conforms to what was suggested in [17].

Table I presents the parameter sensitivity analysis results. We can observe that SaDE is not very sensitive to the parameter choice: medium population sizes (NP: 50, 70, 100) and sufficiently long learning periods (LP: 30, 50, 70) can always lead to the statistically significantly better performance at almost any tested problem dimension size. This finding supports the related claims in the original SaDE work [10] in a statistically rigorous way. It is worth noting that the previously suggested parameter setting (NP:50, LP:50) is also included in the advocated list.

Table II shows SaDE's performance using one superior parameter setting (NP:50, LP:50) suggested by the above parameter sensitivity analysis (also by the original SaDE work [10]). The computational complexity and ECDFs corresponding to such a parameter setting are reported in Table IV and illustrated in Figure 1, respectively. We can observe that SaDE's performance goes down as increasing the problem dimension size. For 10D problems, SaDE has non-zero SRs on 10 out of 28 test functions, i.e., f_1, f_2, f_3, f_4, f_5, f_6, f_{10}, f_{11}, f_{14} and f_{22}. On six out of the remaining 18 test functions, i.e., f_7, f_9, f_{12}, f_{13}, f_{16} and f_{19}, SaDE reaches FEVs less than 1.00e+00 in at least one run. For 30D problems, SaDE has the non-zero SRs on four out of 28 test functions, i.e., f_1, f_5, f_{11} and f_{14}. Among the remaining 24 test functions, SaDE reaches FEVs less than 1.00e+00 in at least one run on two functions, i.e., f_6 and f_{10}. For 50D problems, SaDE has the non-zero SRs on 3 out of 28 test functions, i.e., f_1, f_5 and f_{11}. Among the remaining 25 test functions, SaDE reaches FEVs less than 1.00e+00 in at least one run on two functions, i.e., f_{10} and f_{14}.

Figure 1 illustrates, for each tested problem dimension size, the ECDF of the number of executed function evaluations when the algorithm reaches the pre-specified target FEVs divided by the dimension size and the ECDF of the best achieved FEVs at execution termination divided by the pre-specified target FEVs, which are accumulated over 51 runs of all CEC-2013 test functions. We can observe that, as for smaller target FEVs (10^{-1}, 10^{-4} and 10^{-8}), the proportion of successful execution runs increases from zero at about $316 \cdot D$ function evaluations with respect to any tested dimension size. It is worth noting that when the maxFEvals is reached, the proportion still tends to increase, which means that SaDE may

achieve even better performance if maxFEvals is increased.

We compare the performance of SaDE using the parameter setting (NP:50 and LP:50) as advocated by parameter sensitivity analysis with that of DE/rand/1/bin using the parameter setting (NP:50, CR:0.9 and F:0.5) as typically advised in previous works by applying the Wilcoxcon's signed rank test [18] over all 51 runs of all 28 functions. Table III shows that SaDE achieves the statistically significantly better performance than that of DE/rand/1/bin at any tested dimension size.

V. Conclusions

This paper investigated a well-known DE variant (SaDE) on the CEC-2013 real-parameter single-objective optimization testbed. We carried out parameter sensitive analysis using advanced statistical hypothesis testing methods to compare 30 potentially effective parameter settings over 28 CEC-2013 test functions at 10D, 30D and 50D, respectively. The results reveal that SaDE is less sensitive to the choice of parameters since, i.e., under quite a number of parameter settings SaDE can exhibit the statistically significantly better performance than the other settings, which supports the related claims in the original SaDE work in a statistically rigorous manner. We reported the performance of SaDE using one superior parameter setting suggested by parameter sensitivity analysis according to the protocol of the CEC-2013 testbed along with some additional indices and figures to better depict SaDE's performance. Furthermore, we statistically compared this performance with that of a widely used classic DE strategy DE/rand/1/bin under a typically advised parameter setting, which significantly favors SaDE.

Acknowledgment

This work was supported by NSFC under Grant No. 61005051, SRFDP under Grant No. 20100092120027 and ARC Discovery Grant (DP120102205).

References

[1] R. M. Storn and K. V. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," International Computer Science Institute, Berkeley, CA, USA, ICSI Technical Report 95-012, 1995.

[2] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution - A Practical Approach to Global Optimization*. Springer, 2005.

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

[4] H. A. Abbass, "The self-adaptive pareto differential evolution algorithm," in *Proc.of the 2002 IEEE Congress on Evolutionary Computation (CEC'02)*, vol. 1, Honolulu, Hawaii, USA, 2002, pp. 831–836.

[5] J. Liu and J. A. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Computing*, vol. 9, no. 6, pp. 448–462, Jun. 2004.

[6] S. Das, A. Konar, and U. K. Chakraborty, "Two improved differential evolution schemes for faster global search," in *Proc. of the 2005 conference on Genetic and evolutionary computation (GECCO'05)*. ACM, 2005, pp. 991–998.

[7] M. Omran, A. Salman, and A. Engelbrecht, "Self-adaptive differential evolution," *Computational intelligence and security*, pp. 192–199, 2005.

[8] 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, Dec. 2006.

[9] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. of the 2005 IEEE Congress on Evolutionary Computation (CEC'05)*. IEEE, 2005, pp. 1785–1791.

[10] 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.

[11] Z. Yang, K. Tang, and X. Yao, "Scalability of generalized adaptive differential evolution for large-scale continuous optimization," *Soft Computing*, vol. 15, no. 11, pp. 2141–2155, Sep. 2010.

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

[13] 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, Mar. 2011.

[14] D. Zaharie, "Critical values for the control parameters of differential evolution algorithms," in *Proc. of MENDEL 2002: the 8th International Conference on Soft Computing*, Jun. 2002, pp. 62–67.

[15] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Proc. of the 2008 IEEE Congress on Evolutionary Computation (CEC'08)*. IEEE, 2008, pp. 1110–1116.

[16] 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 cec2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

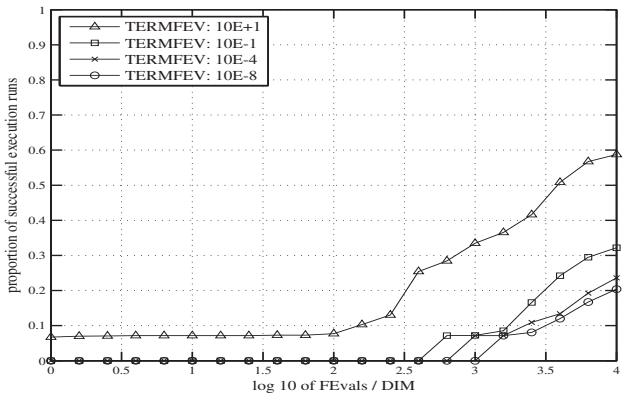
[17] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[18] D. Sheskin, *Handbook of parametric and nonparametric statistical procedures*. CRC PressI Llc, 2004.

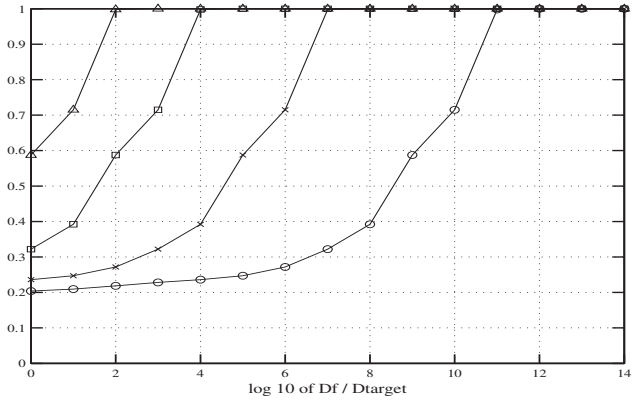
[19] J. J. Liang, B.-Y. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization," Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report 201212, 2013.

[20] K. V. Price, "Differential evolution vs. the functions of the 2^{nd} ICEO," in *Proc. of the 1997 IEEE Congress on Evolutionary Computation (CEC'97)*. IEEE, 1997, pp. 153–157.

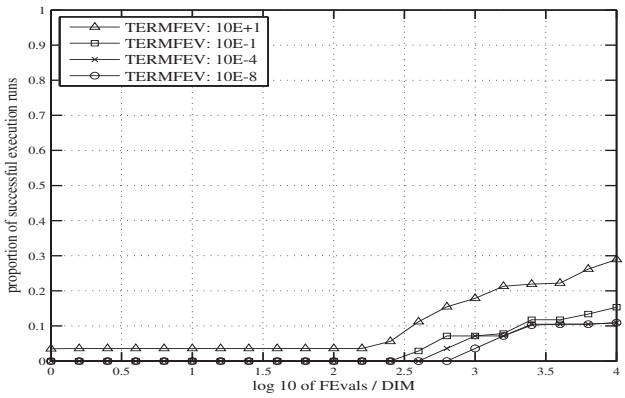
[21] N. Hansen, A. Auger, S. Finck, and R. Ros, "Real-parameter black-box optimization benchmarking: Experimental setup," INRIA, France, Technical Report, 2013.



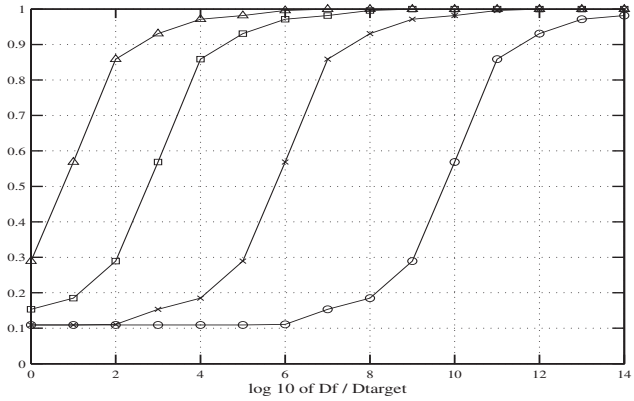
(a) Problem dimension size: 10D



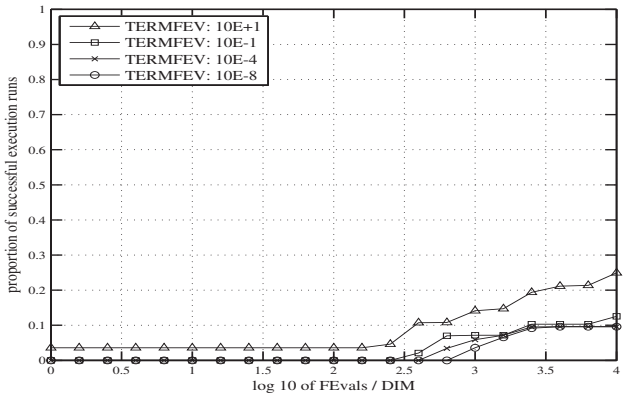
(b) Problem dimension size: 10D



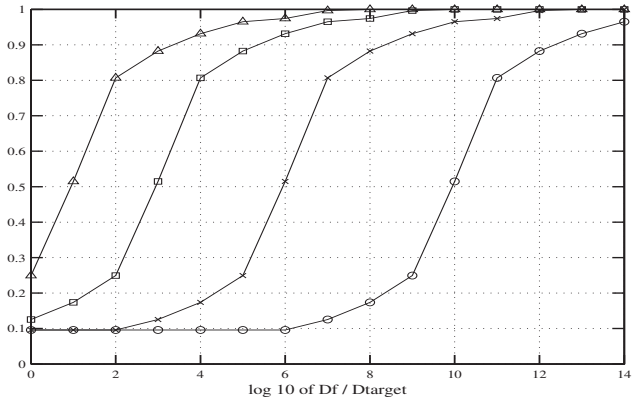
(c) Problem dimension size: 30D



(d) Problem dimension size: 30D



(e) Problem dimension size: 50D



(f) Problem dimension size: 50D

Fig. 1. Empirical cumulative distribution function (ECDF) [21] of **(a)(c)(e)**: the number of executed function evaluations (FEvals) when the algorithm succeeds in reaching certain FEVs (denoted in the legend as TERMFEV: 10e+1, 10e-1, 10e-4 and 10e-8) divided by problem dimension sizes, which is accumulated over all 51 execution runs of 28 CEC-2013 test functions at problem dimension sizes 10D, 30D and 50D, respectively. **(b)(d)(f)**: the best achieved FEVs at termination (Df) divided by certain FEVs (denoted in the legend as TERMFEV: 10e+1, 10e-1, 10e-4 and 10e-8), which is accumulated over all 51 execution runs of 28 CEC-2013 test functions at problem dimension sizes 10D, 30D and 50D, respectively.

Algorithm 2 The Original SaDE Algorithm

Input: *NP*, *LP*

Initialize the generation counter $g = 0$, the strategy selection probability $stPb_{k,g} = 1/4, k = 1, \ldots, 4$, and the $CRm_{k,g} = 0.5, k = 1, 2, 3$; Set the success and failure archives to empty
Initialize the population $\mathbf{P}_g$ of *NP* D-dimensional individuals: $\mathbf{P}_g = \{\mathbf{x}_{1,g}, \ldots, \mathbf{x}_{NP,g}\}$ with $\mathbf{x}_{i,g} = \{x_{i,g}^1, \ldots, x_{i,g}^D\}$
Evaluate the objective function value of each individual in $\mathbf{P}_g$, i.e., $f(\mathbf{x}_{i,g}), i = 1, \ldots, NP$
while the predefined termination criteria are not met **do**
  for $i = 1 \rightarrow NP$ **do**
    Select a strategy index k_i in $\{1, 2, 3, 4\}$ based on $stPb_{k,g}, k = 1, \ldots, 4$ using the stochastic universal sampling
    Randomly generate a *F* value according to the normal distribution $rand_n(0.5, 0.3)$
    Randomly select in $\{1, \ldots, NP\}$ five mutually exclusive indices $r_m, m = 1, \ldots, 5$ that are distinct from i
    Generate a mutant vector $\mathbf{v}_{i,g} = \{v_{i,g}^1, \ldots, v_{i,g}^D\}$:
      if (k_i == 1) "DE/rand/1/bin" **then**
        $\mathbf{v}_{i,g} = \mathbf{x}_{r_1,g} + F \cdot (\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g})$
      end if
      if (k_i == 2) "DE/current-to-gbest/1/bin" **then**
        $\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F \cdot (\mathbf{x}_{gbest,g} - \mathbf{x}_{i,g}) + F \cdot (\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g})$
      end if
      if (k_i == 3) "DE/rand/2/bin" **then**
        $\mathbf{v}_{i,g} = \mathbf{x}_{r_1,g} + F \cdot (\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g}) + F \cdot (\mathbf{x}_{r_4,g} - \mathbf{x}_{r_5,g})$
      end if
      if (k_i == 4) "DE/current-to-rand/1" **then**
        $\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + rand_u(0, 1) \cdot (\mathbf{x}_{r_1,g} - \mathbf{x}_{i,g}) + F \cdot (\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g})$
      end if
    Generate a trial vector $\mathbf{u}_{i,g} = \{u_{i,g}^1, \ldots, u_{i,g}^D\}$:
      if (k_i == 1) or (k_i == 2) or (k_i == 3) **then**
        Randomly generate a *CR* value according to the normal distribution $rand_n(CRm_{k_i}, 0.1)$ subjected to $CR \in [0, 1]$
        $j_{rand} = ceil(rand_u(1, D))$
        for $j = 1 \rightarrow D$ **do**
          $$u_{i,g}^j = \begin{cases} v_{i,g}^j & \text{if } rand_u(0,1) \leq CR \text{ or } j = j_{rand} \\ x_{i,g}^j & \text{otherwise} \end{cases}$$
        end for
      else
        $\mathbf{u}_{i,g} = \mathbf{v}_{i,g}$
      end if
    Evaluate the objective function value of the generated trial vector $\mathbf{u}_{i,g}$
    if ($f(\mathbf{u}_{i,g}) \leq f(\mathbf{x}_{i,g})$) **then**
      $\mathbf{x}_{i,g+1} = \mathbf{u}_{i,g}$, and store the tuple (g, k_i, CR) (if k_i == 1, 2 or 3) or (g, k_i) (if k_i == 4) into the success archive
    else
      $\mathbf{x}_{i,g+1} = \mathbf{x}_{i,g}$, and store the tuple (g, k_i) into the failure archive
    end if
  end for
  if ($g \geq LP$) **then**
    if ($g > LP$) **then**
      Remove those tuples with the first elements smaller or equal to $g - LP$ from the success and the failure archives
    end if
    Calculate $S_{k,g}$ and $F_{k,g}$, $k = 1, \ldots, 4$ as the number of tuples having the second elements equal to k in the success and failure archives, respectively
    if ($S_{k,g} + F_{k,g} > 0$) **then**
      $stPb_{k,g} = S_{k,g}/(S_{k,g} + F_{k,g}) + 0.01$
    else
      $stPb_{k,g} = 0.01$
    end if
    $stPb_{k,g} = stPb_{k,g} / \sum_{k=1,\ldots,4} stPb_{k,g}$
    Calculate $CRm_{k,g}, k = 1, 2, 3$ as the median value of the third elements in those tuples in the success archive having the second elements equal to $k, k = 1, 2, 3$
  end if
  Increase the generation counter: $g = g + 1$
end while

NOTE: (1) $rand_u(a, b)$ is a uniform random number generator sampling in $[a, b]$; (2) $rand_n(a, b)$ is a Gaussian random number generator with mean a and standard deviation b; (3) $ceil(c)$ takes on the smallest integer larger than or equal to c.