THE EXPONENTIAL CROSSOVER IN L-SHADE ALGORITHM

Radka Poláková
Centre of Excellence IT4Innovations, Institute for Research and Applications of Fuzzy Modeling, University of Ostrava, Czech Republic
radka.polakova@osu.cz

Abstract Differential evolution is popular and efficient algorithm for global optimization. L-SHADE algorithm is one of the most successful adaptive versions of the algorithm. It uses only binomial crossover. We study employing the exponential crossover in the algorithm. Our tests are carried out on CEC2015 benchmark set for learning-based optimization competition. According to our results, the employing of the exponential crossover together with binomial one into L-SHADE algorithm is beneficial.

Keywords: Binomial crossover, Differential evolution, Evolutionary algorithm, Experimental comparison, Exponential crossover.

1. Introduction

Differential evolution (DE) is one of the most known and used stochastic algorithms for solving of real-parameter optimization tasks. The algorithm was proposed by Storn and Price in 1997 [10]. There are many researchers who are interested in the algorithm, its behavior, and improvement of its performance [3, 8]. Effectiveness of the algorithm depends on values of its parameters and different values of control parameters are often more beneficial in different stage of search process. These facts are the reasons why many adaptive versions of the algorithm were proposed since differential evolution algorithm appeared, e.g., [1, 2, 7, 14, 18, 21]. The algorithm presented by Tanabe and Fukunaga in 2013 called Success-history based adaptive differential evolution with linear reduction of population size algorithm (L-SHADE) [13] is one of the most effective versions of DE up to these days. L-SHADE employs the most common used type of crossover, binomial one. There are some studies in which types of crossover used in DE are studied

and discussed, e.g., [15, 16, 17, 19]. Mentioned studies and the facts claimed in the papers inspired us to study the impact of including the exponential crossover into L-SHADE algorithm.

In the following two sections, the DE algorithm and L-SHADE algorithm are described. In Section 4, two new versions of L-SHADE algorithm are introduced. Carried out experiments are described and results of them are specified and discussed in Section 5. Conclusions are given in the last section.

2. Differential Evolution

Differential evolution algorithm [10] is one of the most known evolutionary algorithms. DE solves global optimization tasks with continuous search space. Let us have a real function $f : S \to R$, $S \subset R^D$, f is objective function and DE's aim is to find global minimum point of f in S, i.e. such point $\mathbf{x}^*$ that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ holds for all $\mathbf{x} \in S$, S is search space, D is problem dimension. DE works with population P of NP-points from search space S. The points are chosen randomly from S with uniform distribution at the beginning of the search process. NP is the size of population P. Population P then evolves generation by generation. A new generation is created in the following way. A new point $\mathbf{y} \in S$, so called trial point, is created for each member $\mathbf{x}_i$ of population P. If $f(\mathbf{y}) \leq f(\mathbf{x}_i)$ holds, $\mathbf{y}$ replaces $\mathbf{x}_i$ in population P. Otherwise $\mathbf{x}_i$ stays to be a member of population P for next generation. A trial point $\mathbf{y}$ is built up by evolutionary operators mutation and crossover from some points of current generation of population P. A combination of a mutation and a crossover is called DE-strategy. An abbreviation DE/a/b/c is commonly used for a DE-strategy, a is mutation, b is the number of difference vectors used in the mutation, and c is employed crossover. The search process is interrupted, when a stopping condition holds, for example when maximal count of objective function evaluations is reached.

A mutant $\mathbf{u}$ is created by operation mutation for a point $\mathbf{x}_i$. There are many types of mutation used in DE. A mutation is abbreviated by a/b, where a is type of mutation and b is a count of difference vectors used in the mutation. The most common mutation is rand/1 (1). The mutant $\mathbf{u}$ is developed from three random points $\mathbf{x}_{r1}$, $\mathbf{x}_{r2}$, $\mathbf{x}_{r3}$ of current generation of P ($r_1 \neq r_2 \neq r_3 \neq i$). Scaling factor $F \in (0, 2]$ is input parameter usually set as $F \in (0, 1]$. Current-to-pbest/1 mutation (2) was proposed by Zhang and Sanderson [21]. The point $\mathbf{x}_{pbest}$ is randomly chosen point from $p \times 100\%$ best points of population P, p is input

parameter, $0 < p < 1$.

$$\mathbf{u} = \mathbf{x}_{r1} + F \times (\mathbf{x}_{r2} - \mathbf{x}_{r3}), \tag{1}$$

$$\mathbf{u} = \mathbf{x}_i + F \times (\mathbf{x}_{pbest} - \mathbf{x}_i) + F \times (\mathbf{x}_{r1} - \mathbf{x}_{r2}). \tag{2}$$

The trial point $\mathbf{y}$ is created from $\mathbf{x}_i$ and mutant $\mathbf{u}$ by a crossover.

Binomial crossover combines coordinates of $\mathbf{x}_i$ with coordinates of mutant $\mathbf{u}$ into trial point $\mathbf{y}$ according to formula (3).

$$y_j = \begin{cases} u_j & \text{if} \quad U_j \leq CR \quad \text{or} \quad j = l \\ x_{ij} & \text{if} \quad U_j > CR \quad \text{and} \quad j \neq l\,, \end{cases} \tag{3}$$

where l is a number randomly chosen from set $\{1, 2, \ldots, D\}$, $U_1, U_2, \ldots, U_D$ are independent random variables uniformly distributed in $[0,\ 1)$. Input parameter $CR \in [0, 1]$ is crossover parameter.

In exponential crossover, L $(1 \leq L \leq D)$ consecutive coordinates are moved from mutant $\mathbf{u}$ into trial point $\mathbf{y}$. A m is randomly chosen with uniform distribution from set $\{1, 2, \ldots, D\}$. Probability of moving of k-coordinate in sequence $\{u_{m+k}\}$, $0 \leq k \leq (L-1)$ into trial point is CR^k. Coordinates, which are not copied into trial point $\mathbf{y}$ from $\mathbf{u}$, are copied from $\mathbf{x}_i$.

Binomial crossover is more often employed type of crossover than exponential one in adaptive versions of DE [1, 7, 9, 12]. The CR parameter influences the number of elements to be put into $\mathbf{y}$ from $\mathbf{x}_i$ and from $\mathbf{u}$ for both mentioned types of crossover. Let p_m is a mutation probability for a coordinate of $\mathbf{x}_i$ to be replaced by respective coordinate of mutant $\mathbf{u}$. Zaharie found [20] that the relation between probability p_m and CR is linear for binomial crossover and strongly non-linear (described by equality (4)) for exponential crossover.

$$CR^D - \ D\ p_m\ CR + \ D\ p_m - 1 = 0. \tag{4}$$

Figure 1 illustrates the relations between CR and p_m for binomial and exponential crossover and differences amongst them for dimension $D = 30$. The DE algorithm is described by pseudo-code in Algorithm 1.

3. L-SHADE Algorithm

The success-history based adaptive differential evolution algorithm with linear reduction of population size (L-SHADE) proposed by Tanabe and Fukunaga in 2014 [13] is based on SHADE algorithm [11, 12].

SHADE algorithm [11, 12] uses DE/current-to-pbest/1/bin DE-strategy, archive A, and an adaptation of control parameters F and CR. It is

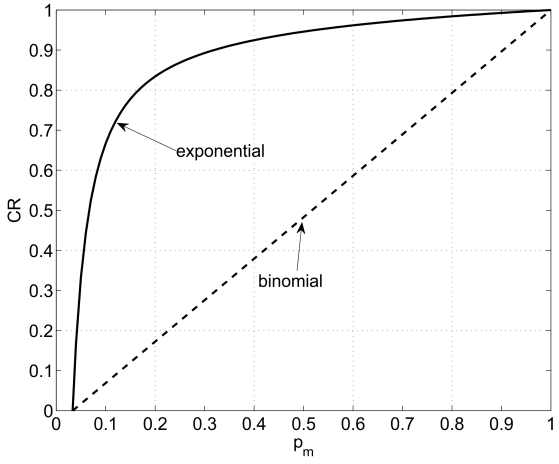

Figure 1: The dependence of CR on probability of mutation p_m for binomial and exponential crossover, $D = 30$

Algorithm 1 Differential evolution algorithm

generate an initial population $P = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{NP})$, distributed uniformly in search space
evaluate $f(\mathbf{x}_i)$, $i = 1, 2, \ldots, NP$
while stopping condition not reached **do**
  $Q := \emptyset$
  for $i = 1$ to NP **do**
    generate a mutant point $\mathbf{u}$ by mutation
    create a trial point $\mathbf{y}$ from mutant $\mathbf{u}$ and $\mathbf{x}_i$ by crossover
    evaluate $f(\mathbf{y})$
    if $f(\mathbf{y}) \leq f(\mathbf{x}_i)$ **then**
      insert $\mathbf{y}$ into new generation Q
    else
      insert $\mathbf{x}_i$ into new generation Q
    end if
  end for
  $P := Q$
end while

based on JADE [21] algorithm. Archive A is initialized as empty set and each point $\mathbf{x}_i$, which is replaced by its better trial point $\mathbf{y}$, is included into archive A during the search process. The archive A is adjusted after each generation to have maximal size of NP, where members for

removing from archive are chosen randomly. $\mathbf{x}_{r1}$ is randomly selected point from P and $\mathbf{x}_{r2}$ is randomly selected point from $P \cup A$ for current-to-pbest/1 mutation (2). The p for the mutation is chosen new for each trial point randomly from interval $p \in (1/NP, 0.2]$ in SHADE algorithm.

F and CR are generated before each new trial point is created from Cauchy and normal distribution, respectively. So called historical circle memories M_F and M_{CR} are implemented for storing several values of the first parameters of F and CR distributions, respectively. Recommended value of the memories' size is $H = NP$ for the SHADE algorithm. Each member of these both memories is set to value 0.5 at the beginning of the search process. The memories are updated in the way described below.

When F and CR are needed for a point $\mathbf{x}_i$ for creation of $\mathbf{y}$, an uniform random number r is chosen from the set $\{1, 2, \ldots, H\}$ and F is random number with Cauchy distribution with parameters $(M_{F_r}, 0.1)$ and CR is random number from normal distribution $N(M_{CR_r}, 0.1)$. If generated F is higher than 1 then F is set to $F = 1$ and if $F \leq 0$ then new value of F is generated. Generated value of CR is trimmed into interval $[0, 1]$, i.e. if $CR > 1$ then $CR = 1$ and if $CR < 0$ then $CR = 0$. If $f(\mathbf{y}) < f(\mathbf{x}_i)$ holds for trial point $\mathbf{y}$ made by the pair of parameters ($\mathbf{y}$ is successful), the F and CR are stored into sets S_F and S_{CR}, respectively. The sets S_F and S_{CR} are set as empty sets at the beginning of a generation.

New values of M_{F_k} and M_{CR_k} (k is current position in M_F and M_{CR}) are computed at the end of the generation from values stored in S_F and S_{CR}, respectively. They are created as weighted means and they are weighted by differences between values of objective function. The M_{F_k} and M_{CR_k} are computed only if there is at least one successful trial point $\mathbf{y}$ in the generation. Then $S_F = \{F_1, F_2, \ldots, F_{|S_F|}\}$ and $S_{CR} = \{CR_1, CR_2, \ldots, CR_{|S_F|}\}$, note that $|S_F| = |S_{CR}|$ holds. The computation of new values M_{F_k} and M_{CR_k} is done according to equalities (5)-(7). $mean_{WL}$ is weighted Lehmer mean and $mean_{WA}$ is weighted arithmetic mean (6). $\mathbf{y}_m$ in (7) is successful trial point generated by F_m and CR_m and $\mathbf{x}_m$ is point of population, which was replaced by trial point $\mathbf{y}_m$.

$$M_{F_k} = mean_{WL}(S_F), \qquad M_{CR_k} = mean_{WA}(S_{CR}), \quad when \;\; S_F \neq \emptyset, \tag{5}$$

$$mean_{WL}(S_F) = \frac{\sum_{m=1}^{|S_F|} w_m \; F_m^2}{\sum_{n=1}^{|S_F|} w_n \; F_n}, \;\; mean_{WA}(S_{CR}) = \sum_{m=1}^{|S_F|} w_m \; CR_m, \tag{6}$$

$$w_m = \frac{\Delta f_m}{\sum_{h=1}^{|S_F|} \Delta f_h}, \quad \Delta f_m = |f(\mathbf{x}_m) - f(\mathbf{y}_m)|, \tag{7}$$

At the beginning of the search process, parameter k is set to $k = 1$. k is increased by 1 after each computation of M_{F_k} and M_{CR_k}. If $k > H$ for such increased k, k is set to $k = 1$.

L-SHADE algorithm [13] is very similar to SHADE algorithm. The only differences are the use of linear reduction of the population size NP and the different setting of some input parameters. The parameter p for current-to-pbest/1 mutation is set on a constant value, $p = 0.11$. The population size is linearly decreasing generation by generation with the increasing number of the objective function evaluations (FES) during the search process from NP^{init} at the beginning to NP^{min} at the end of the search process, i.e. if allowed number of the function evaluations ($MaxFES$) is reached. Relatively big value of NP^{init} is very useful, in order to ensure the most possible exploration of search space. On the other hand, small value of NP^{min} is recommended in order to increase the length of computation (count of created generations) as possible and to let the algorithm to specify the solution as possible. The size of population for generation G+1 is computed according to the formula:

$$NP_{G+1} = round\left[NP^{init} - \frac{FES}{MaxFES}\left(NP^{init} - NP^{min}\right)\right], \qquad (8)$$

where FES is the current number of the objective function evaluations. Whenever $NP_{G+1} < NP_G$, the $(NP_G - NP_{G+1})$ worst individuals are deleted from the population.

Values of the size of archive A, NP^{init}, NP^{min}, and the size of historical circle memories H recommended by authors of [13] are $2.6 \times NP$, $NP^{init} = 18 \times D$, $NP^{min} = 4$, and $H = 6$, respectively. L-SHADE with mentioned parameter setting was the best DE-version in optimization competition on CEC2014 [4, 5]. The L-SHADE algorithm proposed in [13] is described by pseudo-code in Algorithm 2.

4. Exponential Crossover in L-SHADE

In L-SHADE algorithm, each trial point is created from original point $\mathbf{x}_i$ and mutant $\mathbf{u}$ by binomial crossover, the more often used type of crossover in DE. There are studies which focus on the comparison of using of binomial and exponential crossovers in DE in the literature. The influence of employing the exponential crossover in competitive differential evolution adaptive version of DE was studied in [15, 16]. The author found that applying both types of crossover brings improvement for standard functions in comparison with applying only the binomial crossover. For composition functions, the improvement appeared only for part of problems in the study. Tvrdík [17] claimed that the use of both types of crossover together makes DE algorithm more robust.

Algorithm 2 L-SHADE algorithm

1: initialization: NP^{init}, NP^{min}, circle memories M_F and M_{CR}, archive $A = \emptyset$
2: $NP := NP^{init}$
3: generate an initial population $P = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{NP})$
4: evaluate $f(\mathbf{x}_i)$, $i = 1, 2, \ldots, NP$
5: **while** stopping condition not reached **do**
6: set S_F and S_{CR} empty; $Q = \emptyset$
7: **for** $i = 1$ to NP **do**
8: generate F and CR, use circle memories M_F and M_{CR}
9: generate a trial vector $\mathbf{y}$
10: evaluate $f(\mathbf{y})$
11: **if** $f(\mathbf{y}) < f(\mathbf{x}_i)$ **then**
12: save F and CR into S_F and S_{CR}
13: insert $\mathbf{x}_i$ into archive A
14: **end if**
15: **if** $f(\mathbf{y}) \leq f(\mathbf{x}_i)$ **then**
16: insert $\mathbf{y}$ into new generation Q
17: **else**
18: insert $\mathbf{x}_i$ into new generation Q
19: **end if**
20: **end for**
21: $P := Q$
22: modify circle memories if needed, use S_F and S_{CR}
23: $NP_{old} := NP$, recompute size of population NP, eq. (8)
24: **if** $NP < NP_{old}$ **then**
25: remove superfluous points from population
26: **end if**
27: **end while**

Weber and Neri [19] designed a new type of crossover called contiguous crossover. The crossover is similar to exponential crossover and it was shown on a benchmark set that the DE algorithm with the contiguous crossover is either of the same performance or of slightly better performance than DE with binomial crossover in the paper. Based on these works we decided to study a possibility to improve the efficiency of L-SHADE algorithm by employing the other type of crossover, the exponential one, in this paper.

The binomial crossover in L-SHADE can be replaced by exponential crossover or both crossovers could be used together in the algorithm. Proposed versions of L-SHADE algorithm follow:

- L-SHADEexp – the exponential crossover is employed instead of the binomial one,
- L-SHADEcom – both types of crossover compete.

L-SHADEexp version of L-SHADE algorithm employs the exponential crossover instead of the binomial one. In historical circle memories, it stores first parameter of distribution of F and not CR but p_m, the probability which was discussed in Section 2. The memories are labeled M_F and M_{p_m} here. So, when a CR is needed in L-SHADEexp algorithm, it generates p_m from normal distribution $N(M^k_{p_m}, 0.1)$ and then CR is computed from polynomial (4). Each value included in memory M_{p_m} is computed similarly as a value included into memory M_{CR} in L-SHADE, based on successful values of p_m. The other features are the same as for the original L-SHADE algorithm.

Binomial and exponential crossovers compete in L-SHADEcom algorithm. Four historical circle memories, M_{F_b}, M_{CR}, M_{F_e}, and M_{p_m}, and four sets are used for storing the first parameters of distributions and for storing successful values of F and CR of binomial crossover and of F and p_m of exponential crossover, respectively. The crossovers are employed in dependence on probabilities p_b and p_e during the search process.

A crossover is chosen according to current values of probabilities p_b and p_e independently for each point $\mathbf{x}_i$ before its trial point $\mathbf{y}$ is created. Both crossovers have the same probability, $p_b = p_e = 0.5$, at the beginning of the search process. Let s_{b_G} and s_{e_G} are counts of successful trial points generated by DE/current-to-pbest/1/bin and DE/current-to-pbest/1/exp DE-strategy during the generation G, respectively. They are set to $s_{b_G} = s_{e_G} = 0$ at the beginning of each generation. The first change of the probabilities p_b and p_e in the search process and after each resetting of probabilities occurs when a generation, in which the first successful trial point was generated, ends. In such case, at least one of s_{b_G}, s_{e_G} are not equal to zero and probabilities p_b and p_e can be adopted according to (9), (10). The s_b and s_e in (9) and (10) are cumulative counts of successful trial points since the beginning of the search process or since the last reset of probabilities p_b and p_e generated by DE/current-to-pbest/1/exp and DE/current-to-pbest/1/bin DE-strategy, respectively.

$$s_b = s_b + s_{b_G}, \qquad s_e = s_e + s_{e_G}, \tag{9}$$

$$p_b = \frac{s_b}{s_b + s_e}, \qquad p_e = 1 - p_b. \tag{10}$$

If p_b or p_e is less then δ after such re-computation, probabilities p_b and p_e are reset to $p_b = p_e = 0.5$ (δ is input parameter) and also cumulative

Table 1: Improvement and deterioration of L-SHADE algorithm by including exponential crossover

dimension	D=10		D=30		D=50		D=100		all	
algorithm	exp	com	exp	com	exp	com	exp	com	exp	com
improvement	9	7	5	5	3	4	1	2	18	18
deterioration			2	1	4	1	5	2	11	4

counts are reset to $s_b = s_e = 0$. We use value of $\delta = 0.1$ in our experiments. The approach of crossovers' competition used in L-SHADEcom is undertaken from competition of DE-strategies which was proposed for competitive-adaptive version of DE [14].

5. Experiments and Results

Two proposed modifications L-SHADEexp and L-SHADEcom were compared experimentally with L-SHADE algorithm on benchmark set developed for learning-based real-parameter optimization competition on CEC2015 according to conditions defined in [6]. This benchmark set includes 15 different problems of different complexity. Tests were done in dimensions $D = 10, 30, 50, 100$. 51 independent runs were curried out for each function, each dimension, and for each of studied algorithms. The algorithms were stopped according to conditions defined in [6] when *MaxFES* was reached, $MaxFES = D \times 10^4$. Tested algorithms were implemented in Matlab 2010a and this environment was used for experiments. All computations were carried out on a standard PC with Windows 7, Intel(R) Core(TM)i7-4600U CPU, 2.10GHz, 2.70GHz, 8 GB RAM.

The results of our experiments are summarized in Table 2 for $D = 10$ and $D = 30$ and Table 3 for $D = 50$ and $D = 100$. In these tables, best and worst error value, median, mean, and standard deviation of error value of 51 solutions are shown for each benchmark function. The L-SHADE algorithm is referred as *orig*, L-SHADEexp as *exp*, and L-SHADEcom as *com* in the tables. Some results are released from tables. In each released case, the algorithm found the optimum in all runs.

After experiments, we compared statistically results of each of proposed algorithms, L-SHADEexp and L-SHADEcom, with results of original L-SHADE algorithm. Results were compared by Wilcoxon two-sample test. We did the experiments in that way, because we want to

Table 2: Results of L-SHADE versions, D=10 and D=30

	D=10						D=30					
f/Alg.	Best	Worst	Median	Mean	Std	W	Best	Worst	Median	Mean	Std	W
3/orig	1.2410	20.017	20.007	17.599	6.092		20.061	20.147	20.105	20.108	0.0207	
exp	0	20.034	20.001	17.330	6.778	+	20	20.233	20.144	20.123	0.0682	−
com	0	20.020	20.003	16.901	7.275	=	20.002	20.209	20.102	20.107	0.0400	=
4/orig	1.9903	5.9714	3.9800	3.4941	1.002		17.050	31.191	25.097	24.945	3.203	
exp	0.9950	4.9748	2.9849	3.1409	0.8982	+	18.930	33.997	25.880	25.840	3.216	=
com	0	4.9748	2.9849	2.8873	1.076	+	15.931	31.841	24.878	24.461	3.458	=
5/orig	3.7571	141.20	15.539	29.952	38.81		741.77	1720.1	1242.3	1291.4	224.0	
exp	0.24982	229.91	21.825	44.752	54.35	=	1037.8	1899.8	1436.2	1437.0	224.4	−
com	0.18736	137.10	18.597	40.783	45.16	=	911.91	1764.5	1332.7	1345.9	195.6	=
6/orig	0.58746	9.2417	3.2384	3.5703	1.853		36.118	376.13	197.06	205.30	81.01	
exp	0	3.1930	0.4163	0.7743	0.7122	+	32.137	504.58	181.66	194.85	94.57	=
com	0	11.381	1.2031	1.2895	1.680	+	40.539	522.78	188.30	195.37	109.3	=
7/orig	0.06386	0.48082	0.23807	0.24376	0.09122		5.7896	7.4667	6.8496	6.8049	0.3729	
exp	0.02683	0.35244	0.09512	0.12774	0.08196	+	3.9795	7.0493	5.8237	5.7711	0.8452	+
com	0.03655	0.34547	0.12036	0.14226	0.07815	+	4.1992	7.5321	6.5885	6.4734	0.6624	+
8/orig	0.19264	2.9792	0.7688	0.9609	0.6209		15.473	270.83	52.018	55.845	38.11	
exp	5.84E-04	0.8094	0.1052	0.2100	0.2217	+	8.9945	262.85	31.222	42.878	42.21	+
com	4.47E-05	0.6461	0.0348	0.1126	0.1586	+	7.7778	100.43	33.710	38.370	18.72	+
9/orig	100	101.052	100	100.073	0.2523		105.911	108.333	107.076	107.100	0.5177	
exp	100	100	100	100	1.05E-06	+	101.552	106.51	105.908	105.722	0.8040	+
com	100	100	100	100	5.27E-06	+	104.969	106.833	106.175	106.162	0.3326	+
10/orig	140.701	179.010	143.109	148.704	10.88		516.284	741.962	616.434	623.758	56.01	
exp	140.701	152.292	140.708	141.895	2.364	+	533.549	681.669	612.689	604.962	42.55	=
com	140.701	152.292	140.708	142.302	3.429	+	516.315	761.666	614.993	619.535	53.46	=
11/orig	2.1630	4.2013	3.0709	3.0603	0.4503		316.02	623.18	556.30	527.88	82.78	
exp	1.8998	300	2.7454	8.5156	41.63	+	300.39	586.63	418.90	415.99	70.62	+
com	1.5598	301.22	2.8664	8.7553	41.78	=	300.35	613.67	492.17	471.17	93.43	+
12/orig	110.918	112.724	112.166	112.106	0.3543		108.57	110.17	109.36	109.39	0.3637	
exp	109.863	112.135	111.766	111.665	0.3870	+	107.55	109.95	108.95	108.94	0.4547	+
com	111.284	112.445	111.890	111.870	0.2944	+	108.40	110.02	109.17	109.21	0.3842	+
13/orig	0.0925	0.1072	0.0927	0.0940	0.0034		0.0104	0.0109	0.0107	0.0107	1.08E-04	
exp	0.0927	0.1072	0.0927	0.0938	0.0032	=	0.0104	0.0115	0.0107	0.0107	1.99E-04	=
com	0.0925	0.1028	0.0927	0.0942	0.0033	=	0.0104	0.0115	0.0107	0.0107	1.59E-04	=
14/orig	6662.87	6677.01	6670.66	6667.95	4.606		33760	42628	42559	41524	2863	
exp	6662.87	8706.43	6670.66	6707.38	285.6	=	33760	43477	42572	40052	4107	=
com	6662.87	6677.01	6670.66	6668.97	4.616	=	33760	43507	42564	41124	3453	−
15/orig	100	100	100	100	1.04E-13		100	100	100	100	1.54E-13	
exp	100	100	100	100	1.11E-13	=	100	100	100	100	1.15E-13	=
com	100	100	100	100	1.16E-13	=	100	100	100	100	1.28E-13	=

match the comparisons and to know which algorithm (L-SHADEexp or L-SHADEcom) is better compared to original L-SHADE algorithm. The results of curried out statistical tests are depicted in the last column W of Tables 2 and 3. The symbol + means the proposed algorithm reached statistically better results than L-SHADE algorithm. The symbol − means proposed algorithm reached statistically worse results than original algorithm and the symbol = means the null hypothesis of equality of compared results was not reject, which means the compared results are statistically the same. The significance level for all these statistical tests was set to 0.05.

Table 3: Results of L-SHADE versions, D=50 and D=100

	D=50						D=100					
f/Alg.	Best	Worst	Median	Mean	Std	W	Best	Worst	Median	Mean	Std	W
1/orig	78.652	14738	1219.4	2179.8	2739		71420.7	458246	190437	209140	78742	
exp	2.6667	13937	308.18	683.52	1971	+	78354.4	425687	179440	191968	73104	=
com	48.845	6529.6	628.30	1146.0	1371	+	76537.9	446651	180135	196822	73819	=
2/exp	0	0	0	0	0	=	2.07E-08	0.0003	1.75E-06	1.15E-05	3.95E-05	−
3/orig	20.170	20.272	20.228	20.227	0.0253		20.455	20.566	20.507	20.507	0.0248	
exp	20.014	20.363	20.291	20.273	0.0855	−	20.33	20.684	20.607	20.594	0.0682	−
com	20.070	20.309	20.235	20.235	0.0473	−	20.44	20.602	20.531	20.527	0.0343	−
4/orig	38.346	59.506	48.637	49.333	4.645		88.206	127.595	108.236	109.062	8.663	
exp	37.377	70.059	56.652	55.402	6.650	−	104.374	171.201	143.391	145.008	15.07	−
com	32.170	60.880	50.514	50.250	5.762	=	94.656	551.98	120.002	159.407	124.1	−
5/orig	2224.5	3567.9	3052.0	3003.6	284.0		9186.21	11384.5	10640.7	10620.2	495.6	
exp	2579.1	4052.2	3278.0	3296.8	345.3	−	9834.50	13177.3	11706.4	11787.3	794.5	−
com	2212.3	3687.3	2935.1	2995.7	319.1	=	9618.73	11910.4	10728.7	10716.0	496.3	=
6/orig	1108.7	2836.1	1900.9	1936.9	405.9		3421.72	6566.71	5256.00	5153.89	679.0	
exp	783.01	2987.6	1896.5	1926.0	420.2	=	3112.02	6514.91	5084.10	5003.51	722.4	=
com	941.08	2808.0	1762.7	1816.9	397.4	=	3538.54	6507.46	4878.57	4937.39	646.9	+
7/orig	39.677	41.762	40.773	40.725	0.4668		94.895	145.193	138.019	129.696	17.73	
exp	39.620	43.621	40.532	40.591	0.6663	=	96.743	145.853	136.749	124.063	19.89	=
com	38.893	42.902	40.514	40.542	0.7200	=	96.502	147.274	138.728	128.748	18.64	=
8/orig	15.663	758.21	401.87	398.05	178.4		1197.98	3897.73	2567.07	2515.14	631.2	
exp	19.681	916.09	409.65	420.22	210.6	=	1326.36	4183.82	2441.61	2526.49	573.9	=
com	129.57	976.46	395.16	424.10	222.1	=	1215.19	3818.94	2469.14	2427.91	599.0	=
9/orig	102.50	103.09	102.789	102.80	0.1308		110.513	111.694	111.143	111.102	0.3254	
exp	102.15	102.85	102.55	102.52	0.1767	+	107.8	110.451	109.122	109.130	0.5063	+
com	102.43	103.02	102.71	102.71	0.1511	+	109.009	110.974	109.808	109.883	0.4875	+
10/orig	642.96	1689.8	1212.0	1185.9	241.0		3376.98	5300.79	3972.90	3987.88	456.5	
exp	783.37	1765.7	1209.4	1240.7	254.9	=	2888.30	4936.66	3919.99	3874.98	471.00	=
com	791.32	1916.9	1221.5	1233.8	238.0	=	2704.49	4759.48	4001.3	3948.59	451.8	=
11/orig	400	453.65	421.21	416.31	15.78		433.244	655.634	514.352	514.825	41.00	
exp	400	457.41	433.12	424.47	18.62	−	412.963	693.160	560.684	559.924	59.53	−
com	400	473.15	421.21	418.78	19.97	=	447.131	666.403	510.977	526.472	52.46	=
12/orig	115.20	201.54	115.83	127.53	29.81		112.507	200.406	113.395	116.759	17.07	
exp	114.97	201.55	115.68	129.15	31.54	=	112.561	200.409	113.373	118.504	20.68	=
com	115.19	201.54	115.59	122.45	23.32	+	112.476	200.406	113.363	116.721	17.08	=
13/orig	0.0248	0.0256	0.0250	0.0251	1.60E-04		0.0613	0.0658	0.0633	0.0632	9.82E-04	
exp	0.0245	0.0257	0.0250	0.0250	2.62E-04	=	0.0603	0.0661	0.0635	0.0633	0.0014	=
com	0.0246	0.0255	0.0250	0.0250	1.70E-04	+	0.0610	0.0652	0.0633	0.0633	0.001	=
14/orig	52657	52714	52682	52682	15.55		108833	108955	108875	108874	20.46	
exp	52657	52716	52678	52673	15.87	+	108833	108955	108874	108875	19.52	=
com	52657	52727	52681	52680	16.58	=	108855	108910	108871	108873	13.44	=
15/orig	100	100	100	100	1.35E-13		100	100	100	100	1.23E-13	
exp	100	100	100	100	1.24E-13	=	100	100.388	100	100.014	0.0721	=
com	100	100	100	100	1.19E-13	=	100	100.388	100	100.008	0.0547	=

Summarization of statistically significant improvement and deterioration of L-SHADE algorithm by including exponential crossover is depicted in Table 1. The replacement of binomial crossover in L-SHADE algorithm by exponential crossover caused significant improvement for almost two thirds of benchmark functions in dimension $D = 10$, but the count is less for higher dimensions. Additionally, the count of benchmark functions, for which deterioration of results appeared, increases with increasing dimension for the L-SHADEexp algorithm. In case of

including of the crossover competition, L-SHADEcom algorithm, the count of problems, where solution is significantly better than solution of original algorithm, is less than in case of mere replacement of binomial crossover by exponential one in $D = 10$ and for higher dimensions, the count is again less than the count for dimension $D = 10$. However, there is less count of problems, which results are statistically worse than results of L-SHADE algorithm for L-SHADEcom algorithm than for algorithm L-SHADEexp. Thus, we can conclude that including of the exponential crossover into algorithm L-SHADE in the way of competition with binomial one brings better results than only the replacement of binomial crossover by exponential one, which was expected. The algorithm, in which the competition of strategy is employed, can choose appropriate crossover type to solved optimization problem or current stage of process. L-SHADEcom significantly improved the results of L-SHADE in 18 of 60 problems, significant deterioration of results appeared in 4 of 60 tested problems.

Nevertheless, the employing both type of crossover does not increase the performance of L-SHADE algorithm too substantially and does not solve the issue of algorithm's stagnation. The issue of stagnation relates not only to L-SHADE algorithm and our version L-SHADEcom of the algorithm but also to the other adaptive versions of DE. To solve the phenomenon of optimization algorithms' stagnation is not easy task. Discussing adaptive DE-versions, maintenance of population diversity which would be useful for searching different solutions in the search space stands against the convergency of algorithm.

6. Conclusion

The L-SHADE algorithm is successful version of differential evolution algorithm, only the binomial crossover is employed in the algorithm. In this paper, we studied the including of other type of crossover introduced by authors of DE algorithm, the exponential one. We proposed two versions of L-SHADE algorithm in which we used either only exponential crossover or both crossovers together and tested them on CEC2015 benchmark set. According to our results, the employing of the two types of crossover into L-SHADE algorithm is beneficial but it does not solve the issue of DE's stagnation, which remains for further work.

Acknowledgments This work was supported by the project LQ1602 IT4Innovations excellence in science and partially supported by University of Ostrava from the project SGS08/UVAFM/2016.

References

[1] J. Brest, S. Greiner, B. Boškovič, M. Mernik, and V. Žumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Transactions on Evolutionary Computation*, 10:646–657, 2006.

[2] S. Das, A. Ghosh, and S. S. Mullick. A switched parameter differential evolution for large scale global optimization – simpler may be better. *Proceedings of the International Conference on Soft Computing MENDEL*, pages 103–125, 2015.

[3] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15:27–54, 2011.

[4] J. J. Liang, B. Qu, and P. N. Suganthan. Problem definitions and evaluation criteria for the CEC2014 special session and competition on single objective real-parameter numerical optimization. [online] `http://www.ntu.edu.sg/home/epnsugan/`, 2013.

[5] J. J. Liang, B. Qu, and P. N. Suganthan. Ranking results of CEC2014 special session and competition on real-parameter single objective optimization, 2014.

[6] J. J. Liang, B. Y. Qu, and P. N. Suganthan. Problem definition and evaluation criteria for the CEC2015 competition on learning-based real-parameter single objective optimization, 2014.

[7] 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*, 11:1679–1696, 2011.

[8] F. Neri and V. Tirronen. Recent advances in differential evolution: A survey and experimental analysis. *Artificial Intelligence Review*, 33:61–106, 2010.

[9] 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*, 13:398–417, 2009.

[10] R. Storn and K. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[11] R. Tanabe and A. Fukunaga. Evaluating the performance of SHADE on CEC 2013 benchmark problems. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 1952–1959, 2013.

[12] R. Tanabe and A. Fukunaga. Success-history based parameter adaptation for differential evolution. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 71–78, 2013.

[13] R. Tanabe and A. Fukunaga. Improving the search performance of SHADE using linear population size reduction. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 1658–1665, 2014.

[14] J. Tvrdík. Competitive differential evolution. In R. Matoušek and P. Ošmera (Eds.), *Proceedings of the International Conference on Soft Computing MENDEL*, pages 7–12, 2006.

[15] J. Tvrdík. Adaptive differential evolution and exponential crossover. *Proceedings of the International Multiconference on Computer Science and Information Technology (IMCSIT)*, pages 927–931, 2008.

[16] J. Tvrdík. Adaptation in differential evolution: A numerical comparison. *Applied Soft Computing*, 9(3):1149–1155, 2009.

[17] J. Tvrdík. Self-adaptive variants of differential evolution with exponential crossover. *Analele of West University Timisoara, Series Mathematics-Informatics*, 47:151–168, 2009.

[18] J. Tvrdík and R. Poláková. Competitive differential evolution applied to CEC 2013 problems. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 1651–1657, 2013.

[19] M. Weber and F. Neri. Contiguous binomial crossover in differential evolution. *Lecture Notes in Computer Science*, 7269:145–153, 2012.

[20] D. Zaharie. Influence of crossover on the behavior of differential evolution algorithms. *Applied Soft Computing*, 9:1126–1138, 2009.

[21] J. Zhang and A. C. Sanderson. JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, 13:945–958, 2009.