NL-SHADE-RSP Algorithm with Adaptive Archive and Selective Pressure for CEC 2021 Numerical Optimization

1st Vladimir Stanovov
Institute of Informatics and Telecommunication
Reshetnev Siberian State University
of Science and Technology
Krasnoyarsk, Russian Federation
vladimirstanovov@yandex.ru

2nd Shakhnaz Akhmedova
Institute of Informatics and Telecommunication
Reshetnev Siberian State University
of Science and Technology
Krasnoyarsk, Russian Federation
shahnaz@inbox.ru

3rd Eugene Semenkin
Institute of Informatics and Telecommunication
Reshetnev Siberian State University
of Science and Technology
Krasnoyarsk, Russian Federation
eugenesemenkin@yandex.ru

2021 IEEE Congress on Evolutionary Computation (CEC) | 978-1-7281-8393-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/CEC45853.2021.9504959

***Abstract*—This paper proposes a variant of the adaptive differential evolution algorithm, which combines several important concepts, including non-linear population size reduction, rank-based selective pressure in the mutation strategy, adaptive archive set usage, as well as a set of rules to control crossover rate. The developed approach is applied to solve the CEC 2021 Bound Constrained Single Objective Optimization Parametrized Benchmark problems. The performed computational experiments and their statistical analysis show that the proposed NL-SHADE-RSP algorithm is capable of demonstrating high efficiency of fining solutions to biased, shifted and rotated functions compared to other state-of-the-art algorithms, including the winners of the previous competitions.**

***Index Terms*—optimization, differential evolution, selective pressure, CEC 2021**

I. Introduction

The development of efficient single-objective optimization methods represents one of the most important research fields for the Evolutionary Computation (EC), as it allows using these methods in a variety of other approaches. Nowadays, among the variety of heuristic, evolutionary and biology-inspired approaches the Differential Evolution (DE) has proven itself to be capable of outperforming many other approaches in all types of numerical optimization problems [1]. The main properties through which the DE has gained its popularity is its relative simplicity compared to other approaches and high efficiency due to the geometric properties of the applied mutation operators. The integration of parameter control and parameter adaptation techniques has allowed some variants of DE to take leading places in various competitions and find applications in real-world problems, however, there are still possibilities of improvement.

In this paper a new variant of the popular L-SHADE (Linear population size reduction Success History Adaptive DE) is proposed for the IEEE CEC 2021 competition on numerical optimization [2]. The proposed algorithm called NL-SHADE-RSP combines several novel parameter control techniques, such as Non-Linear Population Size Reduction (NLPSR), as well as Linear Crossover Rate Increase (LCRI) and a new strategy adaptation technique, which chooses between two mutation strategies: with and without archive. Other features of NL-SHADE-RSP include usage of Rank-based Selective Pressure (RSP) and linear reduction of p parameter in the current-to-pbest mutation strategy. The experimental results compare the NL-SHADE-RSP with the top 3 algorithms from the CEC 2020 competition, which are IMODE [3], AGSK [4] and j2020 [5], using statistical tests in all predefined scenarios, which include bias, shift and rotation of the goal functions.

The structure of the paper is organized as follows. In Section II a short description of DE in general and L-SHADE algorithm is given. Section III presents the new modifications in detail, and the ideas behind them. The experimental setup and results using CEC 2021 parametrized benchmark problems is presented in Section IV. Finally, Section V contains results discussion and conclusions.

II. Related Work

A. Differential Evolution

The Differential Evolution algorithm was originally proposed for numerical optimization by Storn and Price in [7].

This work was supported by the Ministry of Science and Higher Education of the Russian Federation within limits of state contract № FEFE-2020-0013.

Same as many other Evolutionary Algorithms (EAs), DE maintains a population of solutions during the search process. The key idea of DE is the usage of difference vectors, calculated between the individuals in the current populations. These difference vectors are applied to the members of the population (target solutions) to generate mutant solutions. One of the advantages of DE is small number of parameters: the three main ones are the population size NP, scaling factor for mutation F and crossover probability Cr. The initialization step could be problem-specific, however in most of the cases random generation of points $x_{i,j}$, $i = 1...NP$, $j = 1...D$ in the search space with uniform distribution within $[xmin_j, xmax_j]$ is used, where D is the problem dimension. The original DE used the rand/1 mutation strategy, however most recent DE approaches often use current-to-pbest/1 strategy, originally proposed in JADE [8]. The current-to-pbest strategy works as follows:

$$v_{i,j} = x_{i,j} + F(x_{pbest,j} - x_{i,j}) + F(x_{r1,j} - x_{r2,j}), \quad (1)$$

where $pbest$ is an index of one of the $p*100\%$ best individuals, $r1$, $r2$ are randomly chosen indexes from the population and scaling factor is F usually in the range $[0, 1]$. Indexes $pbest$, $r1$ and $r2$ are generated different from i and each other.

The crossover operation is performed after mutation to generate trial vector by combining the information contained in the target and mutant vectors. The two well-known mutation types are binomial and exponential. In the binomial crossover the trial vector u_i receives randomly chosen components from the mutant vector with probability $Cr \in [0, 1]$ as follows:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } rand(0,1) < Cr \text{ or } j = jrand \\ x_{i,j}, & \text{otherwise} \end{cases}, \quad (2)$$

where $jrand$ is a randomly chosen index from $[1, D]$, required to make sure that the trial vector is different from the target vector to avoid unnecessary fitness calculations. In the exponential crossover first an integer n_1 is chosen randomly in range $[1, D]$ to act as a starting point for crossover, and then the second index n_2 indicating the number of components to be taken from the mutant vector is determined as follows:

$n_2 = 1$
while $rand[0,1] < Cr$ & $n_2 < D$ **do**
  $n_2 = n_2 + 1$
end while

The exponential crossover is then performed using indexes n_1 and n_2 as follows:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } j \in [n_1, n_1 + n_2) \\ x_{i,j}, & \text{otherwise} \end{cases}. \quad (3)$$

After generating the trial vector u the bound constraint handling method (BCHM) is applied. There are many known BCHM [9], and in this study the following approach was used:

$$u_{i,j} = rand(xmin_j, xmax_j) \text{ if } u_{i,j} \notin [xmin_j, xmax_j]. \quad (4)$$

The selection step is performed after calculating the goal function value $f(u)$ in the following way: if the trial vector outperforms or is equal to the parent in terms of fitness, then the target vector x_i in the population is replaced:

$$x_{i,j} = \begin{cases} u_{i,j}, & \text{if } f(u_i) \le f(x_i) \\ x_{i,j}, & \text{if } f(u_i) > f(x_i) \end{cases}. \quad (5)$$

This selection operator is greedy resulting in fitness values in the population either increase or stay the same.

B. L-SHADE algorithm

The development of improved DE variants mainly focused on the parameter adaptation issue, as DE is known to be very sensitive to F, Cr and NP. Modern DE algorithms originate from the JADE [8], followed by SHADE [10] and L-SHADE [11], which took leading positions in CEC competitions. The main features of L-SHADE, which is one of the most popular DE-based optimizers [12] are described below.

The L-SHADE algorithm uses a set of $H = 5$ historical memory cells containing values $(M_{F,k}, M_{Cr,k})$ to generate new parameter values for every mutation and crossover operation. These parameter values are sampled using randomly chosen memory index $k \in [1, H]$ as follows:

$$F = randc(M_{F,k}, 0.1), Cr = randn(M_{Cr,k}, 0.1), \quad (6)$$

where $randc(m, s)$ denotes the random number generated by Cauchy distribution, and $randn(m, s)$ denotes the normally distributed random value, with position parameter m and scale parameter s. The Cr value is set to 0 or 1 if it gets outside the range, and the F value is set to 1 of $F > 1$, however F is generated again if $F < 0$, because $F = 0$ actually cancels mutation operation. The values of F and Cr which delivered an improvement to an individual are saved into arrays S_F and S_{Cr} together with the fitness difference Δf. The memory cell with index h incrementing from 1 to H every generation is updated in the following way:

$$mean_{wL} = \frac{\sum_{j=1}^{|S|} w_j S_j^2}{\sum_{j=1}^{|S|} w_j S_j}, \quad (7)$$

where $w_j = \frac{\Delta f_j}{\sum_{k=1}^{|S|} \Delta f_k}$, $\Delta f_j = |f(u_j) - f(x_j)|$ and S is either S_{Cr} or S_F.The previous parameter values are used to set the new ones with update parameter c as follows:

$$M_{F,k}^{g+1} = c \cdot M_{F,k}^{g} + (1 - c) mean_{wL}(F), \quad (8)$$

$$M_{Cr,k}^{g+1} = c \cdot M_{Cr,k}^{g} + (1 - c) mean_{wL}(Cr), \quad (9)$$

where g is the current generation number. In L-SHADE algorithm the update parameter is set to $c = 0.5$.

The L-SHADE algorithm uses the Linear Population Size Reduction (LPSR) approach to allow broader search at the beginning and better convergence at the end. The population size NP is recalculated at the end of the generation, and the worst individuals are removed. The new number of individuals

follows the linear function depending on available computational resource:

$$NP_{g+1} = round(\frac{NP_{min} - NP_{max}}{NFE_{max}} NFE + NP_{max}), \quad (10)$$

where $NP_{min} = 4$ and NP_{max} are the minimal and initial population sizes, NFE and NFE_{max} are the current and maximal number of function evaluations.

In addition to parameter adaptation techniques, L-SHADE uses an external archive of inferior solutions. The archive A contains parent solutions rejected during selection operation, and is filled until its size reaches the predefined value NA. Once the number of individuals in the archive reaches NA, new solutions replace randomly selected ones in the archive. The current-to-pbest mutation is changed to use the individuals from the archive so that the $r2$ index is taken either from the population or the archive with a probability of 0.5. The archive size decreases together with the population size, following the same LPSR rule.

C. L-SHADE modifications and alternative approaches

The main ideas of the L-SHADE algorithm presented above have led to a variety of similar approaches, each having its own features. The jSO algorithm [13], being the announced winner of the CEC 2017 competition, used several rules to limit the change of F and Cr values, as well as the control of the current-to-pbest stategy greediness. The later was implemented by a linear change of pb parameter as follows: $pb = \frac{pb_{max} - pb_{min}}{NFE_{max}} NFE + pb_{min}$, with the minimal value set to $pb_{min} = pb_{max}/2$ and $pb_{max} = 0.25$.

In 2018 the best non-hybrid DE-based approach, L-SHADE-RSP [14] was ranked second in the competition. The key idea of this method was the implementation of rank-based selective pressure, which was shown to deliver significantly better results for high-dimensional problems. The modified mutation strategy current-to-pbest/r changed the probabilities pr of indexes $r1$ and $r2$ when chosen from the population determined according to the following equation:

$$pr_i = \frac{R_i}{\sum_{j=1}^{NP} R_j}, \quad (11)$$

where ranks $R_i = e^{-i/NP}$ were set as indexes of individuals in an array sorted by fitness values, with largest ranks assigned to best individuals, $i = 1...NP$.

For the CEC 2020 competition on numerical optimization, the top three algorithms were IMODE, AGSK and j2020. The IMODE approach is the L-SHADE like algorithm, which implemented three mutation strategies, namely current-to-pbest with and without archive, and the weighted-rand-to-ϕbest strategy, determined as follows:

$$v_{i,j} = F \cdot x_{i,j} + F(x_{\phi,j} - x_{r3,j}), \quad (12)$$

where ϕ index is chosen from 10% of the best individuals in the population. As long as F is usually sampled between 0 and 1, this is a compressing operator, i.e. it allows simple generation of solutions close to the axis origin. IMODE [3] also uses two crossover types, i.e. binomial with the probability of 0.7 and exponential with the probability of 0.3. The choice between three mutation operators is controlled in a probabilistic manner, where the chance of an operator used for mutation is updated depending on its efficiency in terms of fitness improvement, with a minimum probability set to 0.1 for every operator. Another feature of IMODE is the usage of sequential quadratic programming (SQP) as a local search operator during the last 15% of the computational resource with a probability of 0.1.

The Adaptive Gaining-Sharing Knowledge (AGSK) algorithm [4] implements the concept of knowledge exchange between experienced and non-experienced individuals in the population. The main operators and algorithm structure is still similar to DE, although there is a difference in trial vector generation - the algorithm generates them using two populations of mutant vectors. Also, according to the available source code, AGSK implements non-linear population size reduction as follows:

$$NP_{g+1} = round((NP_{min} - NP_{max}) NFE_r^{1-NFE_r} + NP_{max}), \quad (13)$$

where $NFE_r = \frac{NFE}{NFE_{max}}$ is the ratio of current number of fitness evaluations. AGSK also uses a pool of knowledge factors (similar to F) and knowledge ratios (similar to Cr) and an adaptation procedure to use these variants from this pool.

The j2020 algorithm [5] used two-population design with the large population performing the exploration, and the smaller focusing on exploitation. The parameter update technique used the approach previously proposed in the jDE algorithm, where new F and Cr values were generated with small probabilities, otherwise previously successful values for each individual were used. Also, j2020 made use of algorithm restarts, and used rather small population compared to other approaches, allowing exploration of different areas of the search space.

In [6] the biased Lehmer mean is considered, showing that changing the parameters in eq. (7) may deliver significant performance improvements for a variety of L-SHADE based approaches.

Based on the features introduced in the mentioned algorithms, the NL-SHADE-RSP approach is proposed and described in details in the next section.

III. NL-SHADE-RSP ALGORITHM

The NL-SHADE-RSP algorithm is a further development of L-SHADE-RSP approach, presented in [14] and uses the idea of applying selective pressure, studied in [15]. In NL-SHADE-RSP the same current-to-pbest strategy is used, however, the rank-based selective pressure is applied only to $r2$ index, and only if it is chosen from the population, while $r1$ is chosen uniformly.

The population size is reduced in a non-linear manner, described for AGSK algorithm in the previous section. The graph of population size change example comparing LPSR and NLPSR is presented in Figure 1.

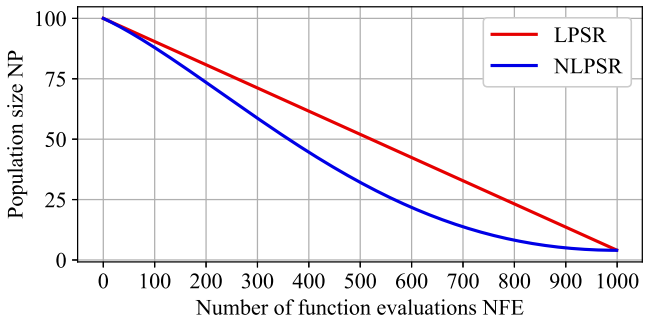

Fig. 1. Linear vs non-linear population size reduction.

As could be seen from Figure 1, NLPSR results in smaller population sizes during the search in case of same initial and final NP settings.

The NL-SHADE-RSP uses automatic tuning of archive usage probability, originated from the strategy adaptation implemented in IMODE algorithm. The probability p_A of archive usage in the last index $r2$ in current-to-pbest strategy is initially set to 0.5, unless the archive is empty. It is then automatically tuned based on the number of usages n_A, which is incremented every time an offspring is generated using archive and sum of fitness improvements achieved with archive Δf_A and without it Δf_P. The archive usage probability is recalculated at the end of each generation as follows:

$$p_A = \frac{\Delta f_A / n_A}{\Delta f_A / n_A + \Delta f_p / (1 - n_A)}. \tag{14}$$

After this the probability p_A is checked to be within [0.1, 0.9] by applying the following rule: $p_A = min(0.9, max(0.1, p_A))$, similar to the rule used in IMODE.

In NL-SHADE-RSP both the exponential and binomial crossovers are used with probability of 0.5. However, different parameter adaptation rules are applied for each of them. For exponential crossover the Success-History Adaptation (SHA) is applied, with the only difference that at the beginning of the generation the crossover rates Cr_i generated for every individual i are sorted according to fitness values, so that smaller Cr are assigned to better individuals. It results in several important effects, first, better individuals receive smaller changes, i.e. most of the times only one component of vector is changed. Second, worse individuals usually tend to receive larger and more often fitness improvements, resulting in larger Cr values adapted by SHA. For the binomial crossover the following parameter control scheme is applied:

$$Cr_b = \begin{cases} 0, & \text{if } NFE < 0.5 NFE_{max} \\ 2\frac{NFE - 0.5}{NFE_{max}}, & \text{otherwise} \end{cases}. \tag{15}$$

In other words, for binomial crossover the Cr_b value is set to 0 for the first half of the search, and then linearly increases to 1 by the end of the computational resource. Note that the Cr_b values do not participate in the parameter adaptation, instead the Cr values sampled by normal distribution are used. This parameter control technique allows improving the search efficiency in cases if the optimized function is non-separable.

The pb value for current-to-pbest mutation in NL-SHADE-RSP is controlled in a similar manner to the jSO algorithm, with the initial pb_{max} set to 0.4 and the final $pb_{min} = 0.2$. The same linear reduction of pb parameter is used, allowing wider search at the beginning and better convergence at the end.

The pseudocode of the NL-SHADE-RSP algorithm is presented in Algorithm 1. The next section contains the experimental setup, parameter settings, results and discussion.

IV. Experimental Setup and Results

The experiments performed in this study followed the rules set by the IEEE Congress on Evolutionary Computation 2021 Competition on Single Objective Bound Constrained Numerical Optimization [2]. The benchmark contained 10 functions defined for $D = 10$ and $D = 20$, and the computational resource is set to $2 \cdot 10^5$ and 10^6 respectively. The main feature of this competition is the parametrized benchmark, which contains 8 variants of transformations, such as bias, rotation and shift, with the main goal to test the effect of possible combinations of transformations. For every benchmark and every function there were 30 independent runs performed, and the achieved best goal function values were saved after $D^{\frac{k}{5}-3} NFE_{max}$ function evaluations, $k = 0, ..., 15$. The algorithm was implemented in C++, compiled with GCC and ran on PC under Ubuntu Linux 20.04, with AMD Ryzen 3700X processor and 48GB RAM, post-processing of results and statistical tests were implemented in Python 3.6.

For comparison top three algorithms of CEC 2020 were used, namely IMODE, AGSK and j2020. All parameters in the source codes provided by the authors were not changed. First, the Mann-Whitney rank sum test with $p = 0.01$ was used to compare NL-SHADE-RSP with competitors, the results for $10D$ and $20D$ are presented in Tables I and II.

TABLE I
Mann-Whitney tests of NL-SHADE-RSP against other approaches, $10D$

Benchmark (code)	IMODE	AGSK	j2020
Basic (000)	0+/6=/4-	4+/6=/0-	5+/5=/0-
Bias (100)	0+/6=/4-	4+/6=/0-	5+/5=/0-
Shift (010)	1+/8=/1-	3+/6=/1-	2+/7=/1-
Rotation (001)	0+/2=/8-	5+/4=/1-	5+/4=/1-
Bias, Shift (110)	1+/7=/2-	5+/4=/1-	4+/5=/1-
Bias, Rotation (101)	0+/2=/8-	6+/3=/1-	3+/6=/1-
Shift, Rotation (011)	6+/4=/0-	7+/3=/0-	3+/6=/1-
Bias, Shift, Rotation (111)	5+/4=/1-	7+/3=/0-	3+/4=/3-
Total	13+/39=/28-	41+/35=/4-	30+/42=/8-

In Tables I and II for every benchmark the summed results of statistical tests over all 10 test functions are given. Here "+" means that NL-SHADE-RSP was significantly better than competitor, "-" means that it was significantly worse, and "=" means that the difference was insignificant. Compared to AGSK and j2020, NL-SHADE-RSP shows superior performance for most problems, except for functions 8 and 10

Algorithm 1 NL-SHADE-RSP

Set $NP_{max} = 30D$, $NP = NP_{max}$ D, NFE_{max},
$H = 20D$, $A = \emptyset$, $M_{F,r} = 0.2$, $M_{Cr,r} = 0.2$, $k = 1$
$NA = 2.1NP$, $n_A = 0.5$, $g = 0$
Initialize population $P_0 = (x_{1,j}, ..., x_{NP,j})$ randomly
while $NFE < NFE_{max}$ **do**
  $S_F = \emptyset$, $S_{Cr} = \emptyset$, $n_A = 0$
  Rank population according to fitness $f(x_i)$
  for $i = 1$ to NP **do**
    Current memory index $r = randInt[1, H + 1]$
    Crossover rates $Cr_i = randn(M_{Cr,r}, 0.1)$
    $Cr_i = min(1, max(0, Cr))$
    repeat
      $F_i = randc(M_{F,r}, 0.1)$
    until $F_i \geq 0$
    $F_i = min(1, F_i)$
  end for
  Sort Cr_i according to fitness $f(x_i)$
  for $i = 1$ to NP **do**
    repeat
      $pbest = randInt(1, NP * p)$
      $r1 = randInt(1, NP)$
      if $rand[0, 1] < p_A$ **then**
        $r2 = randInt(1, NP)$
      else
        $r2 = randInt(1, NA)$
      end if
    until $i \neq pbest \neq r1 \neq r2$
    for j=1 to D **do**
      $v_{i,j} = x_{i,j} + F(x_{pbest,j} - x_{i,j}) + F(x_{r1,j} - x_{r2,j})$
    end for
    Update Cr_b
    if $rand[0, 1] < 0.5$ **then**
      Binomial crossover with Cr_b
    else
      Exponential crossover with Cr_i
    end if
    Calculate $f(u_i)$
    if $f(u_i) < f(x_i)$ **then**
      $x_i \rightarrow A_{randInt[1,NA]}$, $x_i = u_i$
      $F \rightarrow S_F$, $Cr \rightarrow S_{Cr}$, $\Delta f_i = f(x_i) - f(u_i)$
    end if
  end for
  Get NP_{g+1} and NA_{g+1} with NLPSR
  if $|A| > NA_{g+1}$ **then**
    Remove random individuals from the archive
  end if
  if $NP_g > NP_{g+1}$ **then**
    Remove worst individuals from the population
  end if
  Update $M_{F,k}$, $M_{Cr,k}$, p_A
  $k = mod(k, H) + 1$, $g = g + 1$
end while
Return best solution x_{best}

TABLE II
MANN-WHITNEY TESTS OF NL-SHADE-RSP AGAINST OTHER APPROACHES, $20D$

Benchmark (code)	IMODE	AGSK	j2020
Basic (000)	0+/7=/3-	4+/6=/0-	5+/5=/0-
Bias (100)	0+/6=/4-	4+/6=/0-	4+/6=/0-
Shift (010)	6+/4=/0-	6+/4=/0-	5+/5=/0-
Rotation (001)	1+/6=/3-	7+/2=/1-	6+/3=/1-
Bias, Shift (110)	5+/5=/0-	6+/4=/0-	4+/6=/0-
Bias, Rotation (101)	1+/6=/3-	7+/2=/1-	5+/4=/1-
Shift, Rotation (011)	8+/1=/1-	7+/3=/0-	5+/5=/0-
Bias, Shift, Rotation (111)	5+/4=/1-	6+/4=/0-	4+/5=/1-
Total	26+/39=/15-	47+/31=/2-	38+/39=/3-

(composition functions). Compared to IMODE, NL-SHADE-RSP performs worse in $10D$ if all benchmarks are considered, and better in $20D$. For both $10D$ and $20D$ NL-SHADE-RSP mostly wins in cases when there is a function shift present in the benchmark.

Table III shows the Friedman ranking results for all four compared algorithms for $10D$ and $20D$.

TABLE III
FRIEDMAN RANKING OF NL-SHADE-RSP AGAINST OTHER APPROACHES

$10D$				
Benchmark code	NL-SHADE-RSP	IMODE	AGSK	j2020
000	7.892	5.098	11.618	10.686
100	7.873	5.078	11.716	10.627
010	7.686	8.098	10.039	9.471
001	9.029	2.118	12.843	11.304
110	7.912	7.078	10.235	10.069
101	9.147	2.216	13.078	10.853
011	5.206	11.245	10.824	8.020
111	6.167	10.765	11.069	7.294
Total	60.912	51.696	91.422	78.324
$20D$				
Benchmark code	NL-SHADE-RSP	IMODE	AGSK	j2020
000	7.461	5.265	11.245	11.324
100	7.284	5.363	11.196	11.451
010	4.569	11.020	11.186	8.520
001	6.676	6.333	11.922	10.363
110	4.637	10.716	11.569	8.373
101	6.618	6.520	11.863	10.294
011	4.127	12.686	9.990	8.490
111	5.176	10.490	11.078	8.549
Total	46.549	68.392	90.049	77.363

The ranking of results of 30 runs by every algorithm was performed for every function independently, and then all ranks were summed for every benchmark. Table III shows that in most cases IMODE is able to demonstrate better performance than NL-SHADE-RSP for all cases except three: shift; shift, rotation; shift, rotation and bias. Also, worst results were achieved by IMODE in cases where the shift operation was applied. For $20D$ problems in Table III NL-SHADE-RSP has better performance for all benchmarks except basic and biased functions, and similar performance for rotated and biased, rotated functions. The j2020 algorithm shows much better results. Same as before, IMODE shows worse results for shifted benchmarks.

In addition to the statistical tests, the comparison was

performed using the evaluation criteria, provided by the competition organizers [2]. In particular, normalized error values SNE and rankings SR were used to calculate $Score1$ and $Score2$, which together resulted in the final $Score$, assigned to every algorithm. Table IV shows the calculated values of all these metrics.

TABLE IV
ALGORITHMS' SCORES ACCORDING TO CEC 2021 EVALUATION

Metric	NL-SHADE-RSP	IMODE	AGSK	j2020
SNE	8.543	11.242	18.579	10.886
SR	82.250	95.750	157.500	144.500
$Score1$	50.000	37.998	22.991	39.240
$Score2$	50.000	42.950	26.111	28.460
$Score$	100.000	80.948	49.103	67.700

As could be seen from Table IV, NL-SHADE-RSP was able to get best normalized error results and best ranking, outperforming j2020 and IMODE in this cases respectively.

The main idea of the IEEE CEC 2021 competition is to test the algorithms in different conditions and highlight the advantages and disadvantages of every approach. The performed experiments have shown that although the proposed NL-SHADE-RSP algorithm has high efficiency compared to other approaches, in several cases the winner of the CEC 2020, IMODE algorithm has shown much better results. The reason for this difference in performance could be the weighted-rand-to-ϕbest mutation strategy used in IMODE. This strategy allows very simple convergence to the axis origin simply by setting F equal to 0 or very close to this number (as $F = 0$ is usually prohibited in DE). As long as half of the benchmark functions in CEC 2021 do not have shift operation applied, this mutation strategy allows IMODE to easily converge to the global optimum in these cases. For a better demonstration of this effect, another series of experiments has been performed to compare NL-SHADE-RSP and IMODE with a large shift of all functions in all benchmarks, in particular, all functions were additionally shifted to 10000 along all axis for both $10D$ and $20D$. Table V provides the Mann-Whitney test results of NL-SHADE-RSP and IMODE with and without this large shift.

Table VI shows the Friedman ranking of NL-SHADE-RSP, NL-SHADE-RSP with large shift to 10000 (given as NL-SHADE-RSP$_s$), IMODE and IMODE with large shift (IMODE$_s$).

The results in Tables V and VI show that the performance of NL-SHADE-RSP almost does not change even if the function is shifted far away from the axis origin, however, the situation is different for IMODE. For $10D$ functions the IMODE variant that solved problems without large shift has shown better performance in 32 cases and worse only in 6 cases, and for $20D$ functions there were 29 improvements and 13 efficiency deteriorations. Better performance of IMODE that solved problems with large shift was observed in cases where some shift have already been applied according to the competition rules, and the best performance difference was observed for rotated problems, as they are much more difficult to solve if shifted.

TABLE V
MANN-WHITNEY TESTS OF NL-SHADE-RSP AND IMODE AGAINST LARGE SHIFT, $10D$

$10D$		
Benchmark code	NL-SHADE-RSP vs shifted	IMODE vs shifted
000	0+/10=/0-	5+/5=/0-
100	0+/9=/1-	5+/5=/0-
010	0+/10=/0-	1+/8=/1-
001	0+/10=/0-	9+/1=/0-
110	0+/10=/0-	2+/7=/1-
101	0+/10=/0-	9+/1=/0-
011	0+/10=/0-	0+/7=/3-
111	0+/10=/0-	1+/8=/1-
Total	0+/79=/1-	32+/42=/6-
$20D$		
Benchmark code	NL-SHADE-RSP vs shifted	IMODE vs shifted
000	0+/10=/0-	5+/5=/0-
100	0+/10=/0-	5+/5=/0-
010	0+/10=/0-	0+/7=/3-
001	0+/10=/0-	8+/2=/0-
110	0+/10=/0-	0+/8=/2-
101	0+/10=/0-	8+/2=/0-
011	0+/10=/0-	0+/5=/5-
111	0+/10=/0-	3+/4=/3-
Total	0+/80=/0-	29+/38=/13-

TABLE VI
FRIEDMAN RANKING OF NL-SHADE-RSP AND IMODE AGAINST SHIFTED VARIANTS

$10D$				
Code	NL-SHADE-RSP	NL-SHADE-RSP$_s$	IMODE	IMODE$_s$
000	8.961	9.147	5.422	11.765
100	9.333	8.627	5.343	11.990
010	8.265	8.961	8.931	9.137
001	9.657	10.284	1.971	13.382
110	8.637	9.314	8.118	9.225
101	10.078	10.314	2.069	12.833
011	6.304	6.137	12.696	10.157
111	6.833	6.618	11.324	10.520
Total	68.069	69.402	55.873	89.010
$20D$				
Code	NL-SHADE-RSP	NL-SHADE-RSP$_s$	IMODE	IMODE$_s$
000	8.412	8.392	5.382	13.108
100	8.333	8.412	5.608	12.941
010	5.814	6.088	12.608	10.784
001	8.461	8.098	5.706	13.029
110	6.118	6.186	11.814	11.176
101	8.069	8.294	5.716	13.216
011	5.206	4.902	14.167	11.020
111	6.049	5.833	11.412	12.000
Total	56.461	56.206	72.412	97.275

The complexity of the NL-SHADE-RSP algorithm was estimated by calculating T0, T1 and T2 values according to [2]. The results are presented in Table VII.

TABLE VII
COMPUTATIONAL COMPLEXITY OF NL-SHADE-RSP

D	$T0$	$T1$	$T2$	$(T2-T1)/T0$
$D = 10$	3.1E-5	1.6E-5	1.484E-2	4.27097
$D = 20$	3.1E-5	6.5E-5	2.242E-2	5.13548

Tables VIII-XI contain the best, worst, mean, median and standard deviation values achieved by NL-SHADE-RSP.

TABLE VIII
RESULTS FOR $10D$

Function	Best	Worst	Median	Mean	Std
Basic					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	0.0	0.0	0.0	0.0	0.0
f4	9.88E-03	6.22E-02	2.97E-02	3.16E-02	1.50E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	6.84E-03	7.09E-02	4.46E-02	3.90E-02	1.79E-02
f7	9.23E-05	3.72E-02	1.22E-02	1.52E-02	1.03E-02
f8	0.0	0.0	0.0	0.0	0.0
f9	0.0	0.0	0.0	0.0	0.0
f10	2.21E-03	8.85E-03	3.99E-03	4.29E-03	1.51E-03
Bias					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	0.0	0.0	0.0	0.0	0.0
f4	0.0	0.0	2.97E-02	3.10E-02	1.81E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	1.66E-03	7.91E-02	4.38E-02	3.93E-02	2.17E-02
f7	1.46E-03	5.13E-02	2.05E-02	2.27E-02	1.39E-02
f8	0.0	0.0	0.0	0.0	0.0
f9	0.0	0.0	0.0	0.0	0.0
f10	2.77E-03	7.11E-03	3.93E-03	4.14E-03	1.13E-03
Shift					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	2.02E+01	2.02E+01	2.02E+01	2.02E+01	0.0
f4	4.97E-02	2.64E-01	1.09E-01	1.24E-01	5.81E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	3.68E-03	1.07E-01	4.53E-02	4.82E-02	2.66E-02
f7	1.45E-03	6.95E-02	1.85E-02	2.41E-02	1.76E-02
f8	0.0	0.0	0.0	2.08E+01	3.98E+01
f9	1.00E+02	3.00E+02	1.00E+02	1.53E+02	8.84E+01
f10	4.00E+02	4.00E+02	4.00E+02	4.00E+02	0.0
Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	9.37E-02	3.69E+00	4.61E-01	1.09E+00	1.10E+00
f3	0.0	0.0	3.99E-01	5.30E-01	5.72E-01
f4	9.88E-02	7.96E-01	4.89E-01	4.70E-01	1.66E-01
f5	1.20E+00	1.24E+01	4.30E+00	5.19E+00	3.18E+00
f6	6.30E-02	4.78E-01	2.73E-01	2.89E-01	9.81E-02
f7	4.81E-01	9.03E+00	7.67E-01	1.83E+00	2.79E+00
f8	4.90E+02	1.49E+03	1.08E+03	1.09E+03	2.55E+02
f9	0.0	0.0	0.0	0.0	0.0
f10	6.30E+01	6.41E+01	6.35E+01	6.35E+01	2.75E-01
Bias and Shift					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	2.02E+01	2.02E+01	2.02E+01	2.02E+01	0.0
f4	2.96E-02	3.70E-01	1.29E-01	1.53E-01	7.49E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	5.24E-03	1.21E-01	5.61E-02	5.52E-02	2.68E-02
f7	2.70E-03	5.66E-02	2.46E-02	2.55E-02	1.21E-02
f8	0.0	0.0	0.0	2.33E+01	4.23E+01
f9	3.54E+01	3.49E+02	1.00E+02	1.19E+02	6.70E+01
f10	4.00E+02	4.00E+02	4.00E+02	4.00E+02	0.0
Bias and Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	6.25E-02	3.84E+00	1.73E+00	1.22E+00	1.12E+00
f3	0.0	0.0	3.99E-01	4.39E-01	5.84E-01
f4	1.09E-01	7.09E-01	4.77E-01	4.60E-01	1.57E-01
f5	4.19E-01	1.28E+01	4.76E+00	5.59E+00	3.40E+00
f6	7.92E-02	5.70E-01	3.14E-01	3.08E-01	1.12E-01
f7	3.50E-01	1.03E+01	7.81E-01	1.35E+00	2.23E+00
f8	3.45E+02	1.37E+03	9.22E+02	9.14E+02	2.76E+02
f9	0.0	0.0	0.0	0.0	0.0
f10	6.28E+01	6.40E+01	6.33E+01	6.33E+01	3.14E-01

TABLE IX
RESULTS FOR $20D$

Function	Best	Worst	Median	Mean	Std
Basic					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	0.0	0.0	0.0	0.0	0.0
f4	9.88E-03	6.22E-02	2.97E-02	3.16E-02	1.50E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	6.84E-03	7.09E-02	4.46E-02	3.90E-02	1.79E-02
f7	9.23E-05	3.72E-02	1.22E-02	1.52E-02	1.03E-02
f8	0.0	0.0	0.0	0.0	0.0
f9	0.0	0.0	0.0	0.0	0.0
f10	2.21E-03	8.85E-03	3.99E-03	4.29E-03	1.51E-03
Bias					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	0.0	0.0	0.0	0.0	0.0
f4	0.0	0.0	2.97E-02	3.10E-02	1.81E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	1.66E-03	7.91E-02	4.38E-02	3.93E-02	2.17E-02
f7	1.46E-03	5.13E-02	2.05E-02	2.27E-02	1.39E-02
f8	0.0	0.0	0.0	0.0	0.0
f9	0.0	0.0	0.0	0.0	0.0
f10	2.77E-03	7.11E-03	3.93E-03	4.14E-03	1.13E-03
Shift					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	2.02E+01	2.02E+01	2.02E+01	2.02E+01	0.0
f4	4.97E-02	2.64E-01	1.09E-01	1.24E-01	5.81E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	3.68E-03	1.07E-01	4.53E-02	4.82E-02	2.66E-02
f7	1.45E-03	6.95E-02	1.85E-02	2.41E-02	1.76E-02
f8	0.0	0.0	0.0	2.08E+01	3.98E+01
f9	1.00E+02	3.00E+02	1.00E+02	1.53E+02	8.84E+01
f10	4.00E+02	4.00E+02	4.00E+02	4.00E+02	0.0
Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	9.37E-02	3.69E+00	4.61E-01	1.09E+00	1.10E+00
f3	0.0	0.0	3.99E-01	5.30E-01	5.72E-01
f4	9.88E-02	7.96E-01	4.89E-01	4.70E-01	1.66E-01
f5	1.20E+00	1.24E+01	4.30E+00	5.19E+00	3.18E+00
f6	6.30E-02	4.78E-01	2.73E-01	2.89E-01	9.81E-02
f7	4.81E-01	9.03E+00	7.67E-01	1.83E+00	2.79E+00
f8	4.90E+02	1.49E+03	1.08E+03	1.09E+03	2.55E+02
f9	0.0	0.0	0.0	0.0	0.0
f10	6.30E+01	6.41E+01	6.35E+01	6.35E+01	2.75E-01
Bias and Shift					
f1	0.0	0.0	0.0	0.0	0.0
f2	0.0	0.0	0.0	0.0	0.0
f3	2.02E+01	2.02E+01	2.02E+01	2.02E+01	0.0
f4	2.96E-02	3.70E-01	1.29E-01	1.53E-01	7.49E-02
f5	0.0	0.0	0.0	0.0	0.0
f6	5.24E-03	1.21E-01	5.61E-02	5.52E-02	2.68E-02
f7	2.70E-03	5.66E-02	2.46E-02	2.55E-02	1.21E-02
f8	0.0	0.0	0.0	2.33E+01	4.23E+01
f9	3.54E+01	3.49E+02	1.00E+02	1.19E+02	6.70E+01
f10	4.00E+02	4.00E+02	4.00E+02	4.00E+02	0.0
Bias and Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	6.25E-02	3.84E+00	1.73E+00	1.22E+00	1.12E+00
f3	0.0	0.0	3.99E-01	4.39E-01	5.84E-01
f4	1.09E-01	7.09E-01	4.77E-01	4.60E-01	1.57E-01
f5	4.19E-01	1.28E+01	4.76E+00	5.59E+00	3.40E+00
f6	7.92E-02	5.70E-01	3.14E-01	3.08E-01	1.12E-01
f7	3.50E-01	1.03E+01	7.81E-01	1.35E+00	2.23E+00
f8	3.45E+02	1.37E+03	9.22E+02	9.14E+02	2.76E+02
f9	0.0	0.0	0.0	0.0	0.0
f10	6.28E+01	6.40E+01	6.33E+01	6.33E+01	3.14E-01

TABLE X
RESULTS FOR $10D$

Function	Best	Worst	Median	Mean	Std
Shift and Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	6.25E-02	3.75E+00	1.79E+00	1.80E+00	1.25E+00
f3	2.04E+01	2.18E+01	2.08E+01	2.09E+01	3.80E-01
f4	2.08E-01	7.64E-01	5.71E-01	5.59E-01	1.20E-01
f5	4.70E+00	1.47E+02	1.29E+02	1.08E+02	4.82E+01
f6	1.23E-01	7.08E-01	3.24E-01	3.54E-01	1.50E-01
f7	3.73E-01	4.64E+01	8.79E+00	1.01E+01	1.11E+01
f8	2.95E+01	1.00E+02	1.00E+02	9.25E+01	1.57E+01
f9	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.0
f10	3.99E+02	4.14E+02	4.00E+02	4.04E+02	5.94E+00
Bias, Shift and Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	9.37E-02	5.15E+00	1.83E+00	1.84E+00	1.53E+00
f3	2.04E+01	2.15E+01	2.08E+01	2.08E+01	3.47E-01
f4	2.72E-01	8.40E-01	5.63E-01	5.64E-01	1.21E-01
f5	3.09E+00	1.51E+02	1.23E+02	8.83E+01	5.69E+01
f6	1.32E-01	7.62E-01	4.64E-01	4.58E-01	1.68E-01
f7	3.99E-01	3.00E+01	8.84E+00	1.11E+01	1.09E+01
f8	4.07E+01	1.00E+02	1.00E+02	9.36E+01	1.42E+01
f9	1.37E+01	1.01E+02	1.00E+02	9.72E+01	1.55E+01
f10	3.99E+02	4.14E+02	4.01E+02	4.05E+02	5.58E+00

TABLE XI
RESULTS FOR $20D$

Function	Best	Worst	Median	Mean	Std
Shift and Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	6.25E-02	3.75E+00	1.79E+00	1.80E+00	1.25E+00
f3	2.04E+01	2.18E+01	2.08E+01	2.09E+01	3.80E-01
f4	2.08E-01	7.64E-01	5.71E-01	5.59E-01	1.20E-01
f5	4.70E+00	1.47E+02	1.29E+02	1.08E+02	4.82E+01
f6	1.23E-01	7.08E-01	3.24E-01	3.54E-01	1.50E-01
f7	3.73E-01	4.64E+01	8.79E+00	1.01E+01	1.11E+01
f8	2.95E+01	1.00E+02	1.00E+02	9.25E+01	1.57E+01
f9	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.0
f10	3.99E+02	4.14E+02	4.00E+02	4.04E+02	5.94E+00
Bias, Shift and Rotation					
f1	0.0	0.0	0.0	0.0	0.0
f2	9.37E-02	5.15E+00	1.83E+00	1.84E+00	1.53E+00
f3	2.04E+01	2.15E+01	2.08E+01	2.08E+01	3.47E-01
f4	2.72E-01	8.40E-01	5.63E-01	5.64E-01	1.21E-01
f5	3.09E+00	1.51E+02	1.23E+02	8.83E+01	5.69E+01
f6	1.32E-01	7.62E-01	4.64E-01	4.58E-01	1.68E-01
f7	3.99E-01	3.00E+01	8.84E+00	1.11E+01	1.09E+01
f8	4.07E+01	1.00E+02	1.00E+02	9.36E+01	1.42E+01
f9	1.37E+01	1.01E+02	1.00E+02	9.72E+01	1.55E+01
f10	3.99E+02	4.14E+02	4.01E+02	4.05E+02	5.58E+00

V. CONCLUSIONS

The NL-SHADE-RSP algorithm introduced in this paper is based on the L-SHADE and combines recent recently used techniques in Differential Evolution from other approaches. The main differences of NL-SHADE-RSP are adaptive archive usage, rank-based selective pressure, crossover rate parameter control and non-linear population size reduction. The experiments with NL-SHADE-RSP approach have shown that it is highly competitive algorithm compared to the state-of-the art approaches, and it is capable of solving shifted, rotated and biased problems better than other algorithms, especially in high-dimensional search spaces. One of the limitations of the NL-SHADE-RSP is that it could have worse performance compared to IMODE, if the best known solution is located close to the axis origin due to the specific mutation strategy used in IMODE.

REFERENCES

[1] S. Das, S.S. Mullick, P.N. Suganthan, "Recent Advances in Differential Evolution – an Updated Survey", Swarm and Evolutionary Computation, Vol. 27, pp. 1–30, 2016.

[2] Mohamed A.W., Hadi A.A., Mohamed A.K., Agrawal P., Kumar A., P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2021 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization", Technical Report, Nanyang Technological University, Singapore

[3] Sallam, K. M., S. Elsayed, R. K. Chakrabortty, M. Ryan. "Improved Multi-operator Differential Evolution Algorithm for Solving Unconstrained Problems." 2020 IEEE Congress on Evolutionary Computation (CEC) (2020): 1-8.

[4] Mohamed, A. W., A. A. Hadi, A. Mohamed, Noor H. Awad. "Evaluating the Performance of Adaptive GainingSharing Knowledge Based Algorithm on CEC 2020 Benchmark Problems." 2020 IEEE Congress on Evolutionary Computation (CEC) (2020): 1-8.

[5] Brest, J., M. S. Maucec and B. Boskovic. "Differential Evolution Algorithm for Single Objective Bound-Constrained Optimization: Algorithm j2020." 2020 IEEE Congress on Evolutionary Computation (CEC) (2020): 1-8.

[6] Stanovov V., Akhmedova S. and Semenkin E., "Biased parameter adaptation in differential evolution", Information Sciences Vol. 566, pp. 215-238, 2021.

[7] R. Storn and K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, Vol. 11, N. 4, pp. 341-359, 1997, doi: 10.1023/A:1008202821328

[8] J. Zhang, A.C. Sanderson, "JADE: Adaptive differential evolution with optional external archive", IEEE Trans. Evol. Comput., Vol. 13, No. 5, pp. 945–958, 2009.

[9] R. Biedrzycki, J. Arabas, D. Jagodziński, "Bound constraints handling in Differential Evolution: An experimental study", Swarm and Evolutionary Computation, Vol. No. 50, 2019.

[10] R. Tanabe, A. Fukunaga, "Success-History Based Parameter Adaptation for Differential Evolution", Proceedings of the IEEE Congress on Evolutionary Computation, pp. 71–78, 2013.

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

[12] Al-Dabbagh, R. D., Neri, F., Idris, N., and Baba, M. S. Algorithmic design issues in adaptive differential evolution schemes: Review and taxonomy. In Swarm and Evolutionary Computation 43, pp. 284–311 (2018)

[13] J. Brest, M.S. Maucec, B. Boskovic, "Single Objective Real-Parameter Optimization Algorithm jSO", Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1311–1318, 2017.

[14] V. Stanovov, S. Akhmedova and E. Semenkin, "LSHADE Algorithm with Rank-Based Selective Pressure Strategy for Solving CEC 2017 Benchmark Problems," 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, 2018, pp. 1-8. doi: 10.1109/CEC.2018.8477977

[15] V. Stanovov, S. Akhmedova and E. Semenkin, "Selective Pressure Strategy in differential evolution: Exploitation improvement in solving global optimization problems", Swarm and Evolutionary Computation, Volume 50, 2019, ISSN 2210-6502, doi: 10.1016/j.swevo.2018.10.014.