FACTA UNIVERSITATIS (NIŠ)
Ser. Math. Inform. Vol. 38, No 4 (2023), 829–845
https://doi.org/10.22190/FUMI230802054E
Original Scientific Paper

A HYBRID DIFFERENTIAL EVOLUTION FOR NON-SMOOTH OPTIMIZATION PROBLEMS *

Lyudmila D. Egorova[1,2], Lev A. Kazakovtsev[1,2], Vladimir N. Krutikov[1,3], Elena M. Tovbis[2] and Alexandra V. Fedorova[1]

[1]Siberian Federal University, 79 Svobodny Prospekt, 660041 Krasnoyarsk, Russia
[2]Reshetnev Siberian State University of Science and Technology, 31 Krasnoyarskiy Rabochiy Prospekt, 660037 Krasnoyarsk, Russia
[3]Kemerovo State University, 6 Krasnaya Street, 650043 Kemerovo, Russia

Abstract. Solving high dimentional, multimodal, non-smooth global optimization problems faces challenges concerning quality of solution, computational costs or even the impossibility of solving the problem. Evolutionary algorithms, in particular, differential evolution algorithm (DE) proved itself as good method of global optimization. On the other side, approach based on subgradient methods (SG) are good for optimizing non-smooth functions. Combination of these two approaches enables to improve the quality of the algorithm, using the best features of both methods.
In this paper, a new hybrid evolutionary approach (SSGDE) based on differential evolution and subgradient algorithm as the local search procedure is proposed. Behavior of the proposed SSGDE algorithm were studied in a numerical experiment on three groups of generated tests. Comparison of the new hybrid algorithm with the pure DE approach showed the advantage of the SSGDE. The proposed algorithm makes it possible to obtain an improvement in the average best value of the achieved global minimum by three orders of magnitude compared to DE for the non-differentiable test function characterized by high dimensionality, a large number of local extrema, and a significant

Received August 02, 2023, accepted: September 29, 2023
Communicated by Predrag Stanimirović
Corresponding Author: Lev Aleksandrovich Kazakovtsev ()
2010 *Mathematics Subject Classification.* Primary 68T20; Secondary 49J52, 90C59

© 2023 by University of Niš, Serbia | Creative Commons License: CC BY-NC-ND
*This work was supported by the Ministry of Science and Higher Education of the Russian Federation (Grant No.075-15-2022-1121).

ravine. Thus, it has been experimentally established that the proposed method finds the global minimum in the best way for all considered dimensions of the problem with respect to the differential evolution method.

Keywords: global non-smooth optimization, hybrid method, differential evolution, subgradient method.

1. Introduction

Hybridization of algorithms for solving optimization problems is a topic issue, and scientific research in this area is aimed at developing algorithms that have the advantages of combined approaches. The resulting new properties of hybrid algorithms create additional opportunities for their application in various fields to solve global optimization problems [1].

Evolutionary algorithms are often used as part of hybrid algorithms, which is due to the fact that they allow finding good solutions for a large class of complex problems, including those related to multicriteria, multimodal, poorly formalized problems and problems with high dimensionality [2, 3].

The advantages of genetic algorithms (GA) over other optimization methods are related to the fact that they simultaneously analyze different regions of the decision space and find new regions with better objective function values, avoiding the situation of premature convergence at a local minimum [4].

One of the effective population algorithms is the differential evolution algorithm, which was introduced by R. Storn and K. Price [5] for the stochastic population method of global optimization. This method was proposed for solving practical optimization problems with the conditions of finding a true global minimum with fast convergence and the possibility of changing a smaller number of control parameters [5]. This algorithm is used to solve problems in the n-dimensional continuous domain to find the global minimum (maximum) of multimodal, nonlinear, non-differentiable functions.

During the execution of the differential evolution algorithm, a population of candidate solutions is formed, which is improved in an evolutionary way in the process of numerous iterations. Therefore, although the differential evolution algorithm is not biologically determined, it is still classified as an evolutionary computation [2, 6].

Compared to the evolutionary algorithms, the differential evolution method reduces the complexity of genetic operations and enables to dynamically track the search being performed and adjust its strategy. The advantages of the basic DE algorithm are its independence from information about the problem being solved, relative ease of implementation, joint search taking into account both local and group global information, the possibility of using it as part of hybrid algorithms in order to improve performance [7].

The positive results of the DE algorithm have been demonstrated in solving problems characterized by the complexity of equations, the number, variety and variability of the conditions taken into account. The joint use of regular and random processes in the algorithm contributes to finding the global extremum of the function. The algorithm is used to solve optimization problems in the energy sector, related, for example, to the optimal distribution of power units and ensuring reliable power supply to consumers by minimizing the non-admission of electricity during the repair of network equipment [8, 9, 10]. In material science, DE algorithm is used for structure prediction of materials [11]. In engineering, DE is used for feature selection in motor fault diagnosis problem [12], for multiple disk clutch brake design and weight minimization of a speed reducer problems [13]. To optimize the parameters of two models aimed at estimating evapotranspiration, DE and GA proposed in [14].

When solving practical problems of multi-criteria optimization containing conflicting optimality criteria, in order to obtain better results when using the DE algorithm, its modified solutions are developed by selecting parameters, applied mutation operators, schemes for accounting for objective functions, criteria associated with variables and their constraints [15, 16, 17].

A modified solution for the DE algorithm was used in solving problems of optimizing fuzzy rules in self-organizing neural systems when building controllers [18]. Clustering methods are used to extract a set of fuzzy rules, then the DE method is used to tune the parameters of membership functions [19]. The advantages of the multicriteria DE method are presented in relation to the optimization of the parameters of a fuzzy control system and in solving engineering problems [20].

The DE method has demonstrated its effectiveness in solving many practical problems of global multiobjective optimization, but its performance depends on the correct setting of control parameters. For them, there are already proposed values, but when solving specific problems, the need for their individual selection remains relevant [7].

The control parameters of the DE algorithm (crossover rate, scaling factor value) affect the convergence of the algorithm and the convergence rate. To improve the efficiency of the DE algorithm, various solutions are being developed. One of them offers a DE algorithm with a self-adaptive scaling factor and a crossover rate with an elimination mechanism to reduce premature convergence of the algorithm and converge to a local optimum [7].

The development of the practical application of differential evolution algorithms is associated both with the choice of control variables for a certain type of problem being solved, and with hybridization with other optimization methods.

Hybridization of any evolutionary global search algorithm with local search algorithms or with individual learning procedures is one of the developing areas of research on heuristic hybrid global optimization methods represented by memetic algorithms. One of the promising directions for modifying memetic algorithms is hybridization and meta-optimization [21].

The structure of the memetic algorithm is presented by P. Moscato [22]. Memetic algorithms as a wide class of algorithms are represented by the hybridization of evolutionary algorithms (genetic, differential evolution, etc.), local optimization algorithms.

Typical optimization problems encountered in practice, to which hybrid algorithms are successfully applied, are the optimization of the architecture of neural networks, the selection of neural network parameters and weights to achieve the best performance [23], including classification [24], regression, finding optimal parameters for fuzzy control systems [25] and others.

When solving optimization problems in the field of machine learning, non-smooth functions often arise, the lack of differentiability of which creates serious theoretical difficulties. To solve such problems, two main approaches are used: the creation of smooth approximations of non-smooth functions and an approach based on subgradient methods [26, 27]. The efficiency of non-smooth optimization methods in terms of reliability, speed, and accuracy of results is comparable to efficient methods for solving smooth, ill-conditioned problems [28, 29].

When solving problems of minimizing non-differentiable functions, subgradient algorithms with space dilation in the direction of the difference of two successive subgradients have shown good results [28]. To overcome the problem of the convergence rate of subgradient methods, the scientists of the N.Z. Shor group developed families of subgradient methods with space dilation in the direction of the subgradient and in the direction of the difference of two successive subgradients (r-algorithms) [30].

Along with the development of non-smooth optimization methods with space metric transformation in order to increase the speed and quality of convergence of algorithms, as well as to reduce the amount of memory required for their implementation, various modifications of the subgradient method are being developed. In the study of relaxation subgradient methods in [31], an increase in the efficiency of the modified algorithm for non-smooth unconstrained ravine-type optimization problems was demonstrated, but a significant increase in the time costs of the algorithm on problems of large dimension was noted. To overcome this shortcoming and expand the range of nonsmooth high-dimensional problems to be solved [32], an algorithm was proposed with correction of the solution in the iterative process. The algorithm reduces or increases the initial step of one-dimensional descent at iterations depending on the progress obtained, which is determined by the increase coefficient $q_M > 1$ and decrease coefficient $q_m < 1$ specified for the step. A small step decrease rate removes the looping of the method by increasing the neighborhood of the choice of subgradients of the function for solving. The choice of the decrease coefficient value is important for problems of minimizing non-smooth functions, and its value is inversely proportional to the convergence rate of the algorithm. For the best result, this value is recommended to be initially selected experimentally.

In [33], good results are shown for the convergence rate of the relaxation subgradient method with a two-rank correction of the metric matrices on complex non-smooth functions. The use of this algorithm to search for the initial approximation (initial parameters) when training an artificial neural network on small training samples has increased its quality. But it should be noted that the studies carried out in [34] also show results when the successful convergence of gradient methods to a local minimum does not mean that they are able to converge to a global minimum.

Thus, the practical application of subgradient methods for solving minimization problems is associated with an assessment of the effectiveness of their work. This is most relevant in high-dimensional problems for convex functions with different ravines and requires selection of the method parameters. Another area of research is solving the problem of unattainability of the specified stopping criteria by the algorithm or their achievement with significant time costs. Solutions aimed at improving the efficiency of convergence of methods for various applied problems are presented both by modification of optimization algorithms and by the development of hybrid algorithms [35, 36].

The aim of our study is combining the ability of evolutionary algorithm to find a global minimum with the abilities of subgradient methods [37] to search locally for convex non-differentiable functions and smooth functions that difficult to minimize (for example, ravine-type functions [28]), to improve the convergence of the hybrid algorithm on multimodal non-convex functions.

The rest of the paper is organized as follows. In Section 2, the problem of the study is formulated, as long as the method of differential evolution and subgradient method are described. In Section 3, the implementation of proposed algorithm is given. In Section 4, experiments with algorithm were made. Section 5 concludes the work.

2. Problem statement and description of methods

Consider the minimization problem [2]:

$$\min_{x} f(x), \tag{2.1}$$

where $f(x)$ is objective function (cost function), x is the independent variable, which is an n-dimensional vector in continuous space:

$$x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n. \tag{2.2}$$

The number of elements n in the vector x is the dimension of the optimization problem. Define a local minimum x^* as $f(x^*) < f(x)$ for all x, such that:

$$\|x - x^*\| < \varepsilon, \tag{2.3}$$

where $\|x - x^*\|$ is some measure of distance, and $\varepsilon > 0$ is some user-defined neighborhood size. We define the global minimum as:

$$f(x^*) \leq f(x), \tag{2.4}$$

for all x, possibly with constraints on the allowable values of x.

The DE algorithm operates on a population of possible solutions $\mathcal{P}$ of the generation g, each individual solution is an individual of population that is a potential optimal solution [38]:

$$\mathcal{P}^g = \{X_i^g\}, i = 1, \dots, N_p, \tag{2.5}$$

where N_p is population size. The variables that make up an individual are called genes:

$$X_i^g = \{x_{i,j}^g\}, j = 1, \dots, n, \tag{2.6}$$

where n is number of variables (genes) of the individual. In this sense, we can say that the category "vector of variables" plays the same role here as the category "genotype" in biology [39]. The population is initialized randomly, taking into account the existing constraints on the allowed values of x:

$$\mathcal{P}^0 = \{x_{i,j}^0\} = \{rand_{i,j} \cdot (h_j - l_j) + l_j\}, \tag{2.7}$$

where $rand$ is a function that generates random values uniformly distributed in the range [0,1), l_j and h_j are the lower and upper bounds for the element x_j. Then, differentiation and recombination operations are applied to each individual of the population, the main idea of which is to form a candidate solution from the difference of two other random vectors (individuals), then scaling the resulting difference vector and adding it to the third random vector (individual) [2]. Thus, the traditional crossover and mutation operators are not used in the DE algorithm, but specialized operators are used that modify the value of the current individual based on three other random individuals $\pi = \{\xi_1, \xi_2, \xi_3\}$, extracted from the population [40]. The resulting candidate solution is obtained by the formula:

$$\tau = \xi_3 + F \cdot (\xi_2 - \xi_1), \tag{2.8}$$

where $F > 0$ is step size (scaling factor, constant of differentiation) that determines the influence of the difference vector on the mutant vector. Next, the recombination process is performed, in which one of the genes of the trial vector inherits the gene of the current individual with some probability C:

$$w_j = \begin{cases} \tau_j & \text{if} \quad rand_j \leq C \\ ind_j & \text{otherwise}, \end{cases} \tag{2.9}$$

where $j = 1 \dots n$, τ_j is the $j - th$ component of the candidate solution τ, $rand_j$ is random value from uniformly distributed values interval in the range $[0, 1]$, C is crossover probability (constant of crossover), $C \in [0, 1]$, ind_j is $j - th$ component of the current individual. Then, the resulting candidate solution is evaluated based on the value of the cost function. If the value of the cost function for the trial vector is not greater than for the current individual, then it replaces the current individual:

$$ind = \begin{cases} w & \text{if} \quad f(w) \leq f(ind) \\ ind & \text{otherwise}. \end{cases} \tag{2.10}$$

Next, a subgradient algorithm is applied to the individual, described in [26, 27].

According to [41], subgradient of the convex function $f(x)$ at the point $x_0 \in \mathrm{dom} f$ is vector $g(x_0)$ satisfying the condition

$$f(x) - f(x_0) \geq \langle g(x_0), x - x_0 \rangle, \ x \in \mathrm{dom} f. \tag{2.11}$$

Hereinafter, $\langle \cdot, \cdot \rangle$ is a dot product of vectors. The anti-subgradient at the point x_0 forms an acute angle for any direction from the point x_0 to the point x, which has a smaller value $f(x)$. The subgradient method for minimizing non-differentiable functions is based on the fact that if there is a minimum of the function $f(x)$ and the point x_0 is not a minimum point, then when shifting from this point in the direction $-g(x_0)$, the distance to the minimum point is reduced.

The subgradient method is represented by an iterative sequence $\{x_k\}_{k=0}^{\infty}$ built in accordance with the following rule [26]:

$$x_{k+1} = x_k - \gamma_k s_{k+1}, \ \gamma_k = arg \min_{\gamma} f(x_k - \gamma s_{k+1}), \tag{2.12}$$

where k is the iteration number, γ_k is the stepsize, x_0 is a given starting point, and the descent direction s_{k+1} is a solution of a system of inequalities on $s \in \mathbb{R}^n$ [26]:

$$\langle s, g \rangle > 0, \ \forall g \in G. \tag{2.13}$$

Here, G is a set of subgradients calculated on the descent trajectory of the algorithm at a point x_k.

Vector s^* is the solution of the system (2.13). It forms an acute angle with each of the subgradients of the set G. If the subgradients of some neighborhood of the current minimum of (2.12) act as the set G, then iteration (2.12) for $s_k = s^*$ provides the possibility of going beyond this neighborhood with a simultaneous decrease in the function.

The following approach had been proposed in [26] to reduce the system (2.13) to a system of equalities. Let $G \subset R^n$ belongs to some hyperplane, and its vector $\eta(G)$ closest to the origin be also the vector of the hyperplane closest to the origin. In this case, the solution of the system $\langle s, g \rangle = 1, \ \forall g \in G$ is also a solution for (2.13):

$$\langle s, g_i \rangle = y_i, \ i = 0, 1, ..., k, \ y_i \equiv 1. \tag{2.14}$$

Figure 2.1 shows the projection of a subgradient set lying on a straight line in the plane of vectors g_1 and g_2. The vector $\eta(G) \in G$ lies in this plane and is the normal of the hyperplane $\langle s^*, g \rangle = 1$ formed by the vectors g at $s^* = \eta(G)/||\eta(G)||^2$.

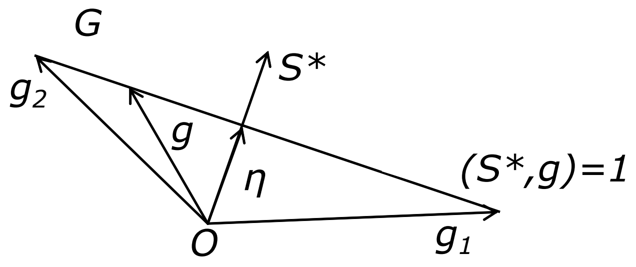

Fig. 2.1: Projection of a subgradient set G

3. Algorithmic implementation

The proposed SSGDE algorithm is based on the classical differential evolution algorithm and the subgradient local search algorithm described in [26, 27]. In accordance with Wang's classification, this hybrid algorithm can be classified as high-level embedded hybridization [3]. The following rules for combining DE and the subgradient algorithm were used in the algorithm: after evaluating a DE candidate solution a subgradient algorithm is applied to the vector x_i, the resulting new candidate solution is evaluated, and if it turns out to be the best, then the current vector x_i is replaced by a new solution. Thus, the DE algorithm works as is, but at the end of each cycle, a subgradient method is applied to a possibly already modified individual, which leads to an improvement in the convergence of DE, due to the fact that each individual reaches a local minimum, in the vicinity of which it is located [3, 42, 43, 44]. The proposed hybrid algorithm is represented by the following pseudocode (3.1):

```
Algorithm 3.1.
Step 1. Set options:
F = step size (scaling factor);
C = crossover probability $\in [0, 1]$;
G = subgradient algoritm running probability $\in [0, 1]$;
$N_p$ = population size;
$N_g$ = number of generation.
Step 2. Initialize the population of candidate solutions $\mathcal{P}^0 \leftarrow \{x_i\}, i \in [1, N_p]$.
Step 3. Calculate the fitness values of all individuals in the population $f(\mathcal{P}^0)$.
Step 4. While not (termination condition) do:
Step 4.1. For each generation $g_k$, $k \in [1, N_g]$, do:
Step 4.1.1 For each individual $x_i$, $i \in [1, N_p]$, do:
Step 4.1.1.1 $r_1 \leftarrow$ choose randomly int $\in [1, N_p]$: $r_1 \notin \{i\}$
Step 4.1.1.2 $r_2 \leftarrow$ choose randomly int $\in [1, N_p]$: $r_2 \notin \{i, r_1\}$
Step 4.1.1.3 $r_3 \leftarrow$ choose randomly int $\in [1, N_p]$: $r_3 \notin \{i, r_1, r_2\}$
Step 4.1.1.4 Create differential mutation vector $v_i \leftarrow x_{r1} + F \cdot (x_{r2} - x_{r3})$
Step 4.1.1.5 $\psi_r \leftarrow$ choose randomly int $\in$ [1, n]
Step 4.1.1.6 For each dimension $j \in$ [1, n], do:
Step 4.1.1.6.1. $r_j \leftarrow$ choose randomly real $\in$ [0, 1)
Step 4.1.1.6.2. If $(r_j \leq C)$ or $(j = \psi_r)$ then $u_{ij} \leftarrow v_{ij}$ else $u_{ij} \leftarrow x_{ij}$ End if
Step 4.1.1.6.3. Next iteration of loop 4.1.1.6 (dimension index)
Step 4.1.1.7. If $f(u_i) < f(x_i)$ then $x_i \leftarrow u_i$
Step 4.1.1.8. $q_i \leftarrow$ choose randomly real $\in [0, 1]$
Step 4.1.1.9. If $(q_i < G)$ then $s_i \leftarrow \text{subgg}(x_i)$ (apply the subgradient algorithm)
Step 4.1.1.9.1 If $f(s_i) < f(x_i)$ then $x_i \leftarrow s_i$
Step 4.1.1.10. Next iteration of loop 4.1.1 (individual index)
Step 4.1.2. Next iteration of loop 4.1. (generation index)
Step 4.2. End while
```

4. Experimental design

To find the optimal settings for the developed SSGDE algorithm, to prove its performance, and to compare the obtained algorithm with the original DE algorithm at the experimental level, a series of computational experiments was carried out. Tests were performed on a computer with "AMD FX(tm)-8320" Eight-Core Processor 3.50 GHz, 14 GB of random access memory and "Windows 10 Pro" operating system. The software implementation of the algorithm was carried out in Delphi.

For each group of tests, 30 Monte Carlo simulations were run, and then the results were averaged. Initialization of the initial population of candidate solutions for the original DE algorithm and the tested SSGDE algorithm (initial approximation of a function) was obtained using a random number generator in the range [-100,100]. Then, for each iteration of the DE algorithm, with a probability G, a subgradient algorithm was launched. The initial approximation of the function being optimized was the coordinates of the current candidate solution of DE after modification and evaluation.

The specific parameters of differential evolution F (step size, scaling factor) and C (crossover probability) for both tested algorithms were chosen using the grid search procedure. For subsequent experiments F = 0.4, C = 0.7. For testing, an artificial problem was used, which contains several difficulties, namely, high dimensionality, a large number of local extrema, and a significant ravine. To find out the influence of these factors on the efficiency of the algorithm, several series of tests were performed for different values of the test function parameters that affect these factors. Below is the analytical expression (4.1) for the test function used in the computational experiments.

$$f(x) = min\left\{\sum_{i=1}^{M}\sum_{j=1}^{n} a_{ij}|x_j - c_{ij}| + b_i\right\}. \tag{4.1}$$

This test function consists of a set of non-differentiable, one-extremal functions, to which the minimum operator is applied. In the formula (4.1) M is the number of component functions, n is the problem dimension. The minima of these functions are at points c_{ij} and take minimal values equal to b_i. The coefficient a_{ij} is the stretch coefficient, which affects the degree of "gullyness" of the ith function along the j coordinate. The landscape of this test function for $n = 2$ is shown in Figure 4.1.

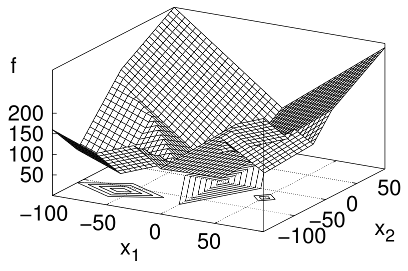

Fig. 4.1: Landscape of a two-dimensional test function (4.1)

The computational complexity of the DE and SSGDE algorithms was measured using the number of computations of the minimized function $ifun$.

4.1. First group of tests

The proposed SSGDE algorithm is a memetic algorithm (MA). Memetic algorithms combine one of the evolutionary global optimization algorithms with one of the local optimization algorithms [4, 45]. In order to trace the influence of the local search provided by the subgradient method on the convergence of the differential evolution algorithm, the first group of computational tests was carried out.

In this group of tests, we studied the dependence of the best approximation to the optimum for the SSGDE algorithm on the limitation on the number of function calculations in the subgradient block of the algorithm. This limitation affects the "depth" of the local search of the subgradient method.

The stopping criteria for this test group were:

For global search:

1. By the value of the function (when the desired accuracy δ of the optimal solution is reached):
   (4.2) $$f(x_k) - f^* \leq \delta,\ \delta = 10^{-5}.$$
2. By the number of generations DE (forced stop if the number of generations exceeds a predetermined number): $N_g = 25$.

For the local search block of the subgradient algorithm:

1. By the number of function evaluations: $ifun_{max}$. This parameter varied from a range of values {1000, 2000, 3000, 4000, 5000}.

Additional fixed parameters of the algorithm used in this group of tests, as well as the results obtained, are shown in the Table 4.1 and in the Figure 4.2.

Table 4.1: Dependence of the efficiency of the SSGDE algorithm on the limitation on the number of function evaluations $ifun_{max}$ in the subgradient method for the test function (4.1) with the following values: the number of dimensions $n = 100$, the number of functions $M = 100$, the stretch coefficient $a_{max} = 1$; population size $N_p = 100$; probability of running subgradient algorithm: $G = 1$.

$ifun_{max}$	$ifun$	fm (min)	$TimeL$ s.
1000	2.51e+06	5.08e-02	2.20e+02
2000	5.01e+06	3.32e-04	4.44e+02
3000	5.93e+06	6.21e-05	5.32e+02
4000	5.92e+06	6.20e-05	5.26e+02
5000	5.92e+06	5.66e-05	5.24e+02

It can be seen from the Table 4.1 and Figure 4.2 that the accuracy of the SSGDE algorithm increases with the number of function calculations in the subgradient method and reaches the desired accuracy at $ifun_{max} \approx 3000$.

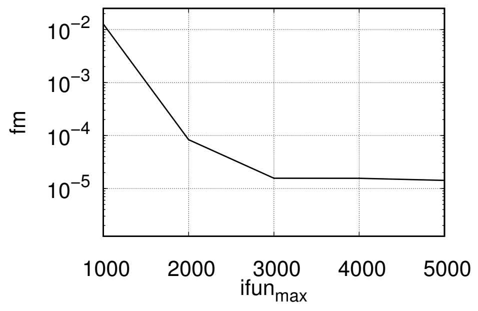

FIG. 4.2: Dependence of the minimum value of the achieved optimum fm on the limit on the number of evaluations of the function $ifun_{max}$ in the block of the subgradient algorithm

4.2. Second group of tests

The second group of tests was carried out in order to trace how the increase in the dimension of the problem n affects the ability of the algorithm to find the global minimum. In this group of tests, n was increased to 1000. The following algorithm termination criteria were applied:

1. By algorithm running time $T_{max} = 600$ sec.
2. By the value of the function (4.2).

For the local search block of the subgradient algorithm:

1. By the number of function evaluations $ifun_{max} = 3000$.

The mean value of the number of evaluations of the function $ifun$ and the gradient ig, as well as the fm (min), fm (mean) and fm (median) value of the reached optimum were calculated over 30 Monte Carlo simulations. For calculations in this group of tests, the following fixed algorithm parameters were used: test function (4.1) with the following values: number of functions $M = 10000$, stretch coefficient $a_{max} = 1$; DE parameters: population size $N_p = 100$, probability of running subgradient algorithm: $G = 1$.

The results are shown in Table 4.2.

Table 4.2: Comparison of DE and SSGDE algorithms

	$ifun$	ig	fm		
Method			min	mean	median
DE	5.99e+03	-	4.54e+04	4.69e+04	4.66e+04
SSGDE	7.47e+03	7.36e+03	2.43e-04	5.15e+01	3.91e+01

Table 4.2 shows that SSGDE performs significantly better than DE. The difference between the average value of the best achieved minimum fm (mean) for the DE and SSGDE algorithm is 3 orders of magnitude, and between the minimum value of the achieved minimum fm (min) is 8 orders of magnitude.

4.3. Third group of tests

In the third group of tests, the number of M functions was increased to 100000. The purpose of this test was to check how the algorithm behaves on high-dimensional problems with a large number of local extrema and with the presence of functions with a high degree of ravine.

The ravine coefficient a_{ij} of the test function (4.1) was varied from the values of the series $\{1, 10, 100, 1000, 10000, 100000\}$. The following algorithm termination criteria were applied:

1. By the value of the function (4.2);
2. By algorithm running time $T_{max} = 1800$ sec.

For the local search block of the subgradient algorithm:

1. By the number of function evaluations $ifun_{max} = 500$.

The results are shown in Table 4.3.

The results of this test show that as the stretch coefficient a_{max} increases by one order, the best accuracy of the achieved minimum fm also decreases by one order, i.e. there is an inverse linear relationship. The graph of this dependence is shown in Figure 4.3. At the same time, the SSGDE algorithm shows three orders of magnitude better accuracy of the achieved minimum of the function fm (min), than the DE algorithm.

Table 4.3: Dependence of the efficiency of the DE and SSGDE algorithms on the value of the stretch coefficient a_{max}. Number of dimensions $n = 1000$, population size $N_p = 100$, probability of running subgradient algorithm: $G = 0.1$

	a_{max}	$ifun$	ig	fm		
Method				min	mean	median
DE	1	1975	-	4.88e+04	5.52e+04	5.46e+04
	10	1967	-	4.86e+05	5.56e+05	5.74e+05
	100	2182	-	4.80e+06	5.44e+06	5.42e+06
	1000	2176	-	4.88e+07	5.47e+07	5.38e+07
	10000	2133	-	4.80e+08	5.41e+08	5.33e+08
	100000	2024	-	4.76e+09	5.41e+09	5.41e+09
SSGDE	1	2082	1938	3.13e+01	8.13e+01	7.63e+01
	10	1984	1843	3.60e+02	8.38e+02	7.56e+02
	100	2022	1882	3.94e+03	6.67e+03	6.43e+03
	1000	2051	1911	3.10e+04	5.26e+04	5.28e+04
	10000	2108	1970	2.96e+05	3.58e+05	3.53e+05
	100000	1963	1828	2.89e+06	3.22e+06	3.21e+06

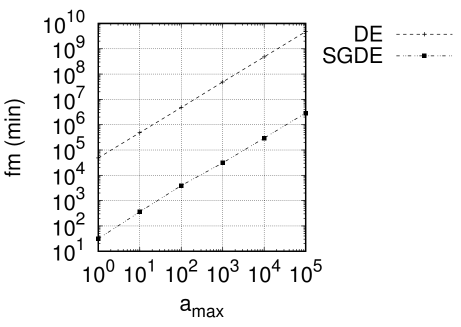

FIG. 4.3: Dependence of the minimum value of the achieved optimum fm (min) on the stretch coefficient a_{max}

5. Conclusion

Differential evolution is an efficient, reliable and simple evolutionary method, however, it has some disadvantages such as slow convergence. Hybridization in this case can contribute to obtaining additional capabilities of the algorithm in solving global optimization problems.

In this study, we propose a new SSGDE algorithm combining differential evolution with properties of subgradient minimization method. Hybridization of DE with subgradient local search makes it possible to achieve a significant improvement in the convergence of the optimization algorithm for large-dimensional, multimodal, non-convex, non-differentiable, ravine-type functions.

Computational experiments on three groups of tests were conducted, in which the influence of the dimension of the problem, the number of function evaluations, and the stretch parameter on the efficiency of the algorithm was studied. The best approximation was achieved for a stretch parameter equal to 1. In addition, the proposed algorithm was compared with the differential evolution algorithm. Experimental results confirm the efficiency of the new algorithm.

For further conclusions about the choice of optimal settings for SSGDE algorithm when solving various optimization problems, including global multi-parameter optimization, additional studies of the algorithm's efficiency with changing parameter values for problems of equal high dimension and with other functions are required.

REFERENCES

1. M. Oszust, G. Sroka and K. Cymerys: *A hybridization approach with predicted solution candidates for improving population-based optimization algorithms.* Information Sciences. **574(3)** (2021), 133–161.
2. D. Simon: *Evolutionary Optimization Algorithms. Biologically-Inspired and Population-Based Approaches to Computer Intelligence.* Wiley, 2013.
3. X.-S. Yang: *Nature-Inspired Optimization Algorithms.* Elsevier, London, 2014.
4. P. Garg: *A Comparison between Memetic algorithm and Genetic algorithm for the cryptanalysis of Simplified Data Encryption Standard algorithm.* IJNSA. **1(1)** (2009), 34 - 42.
5. R. Storn and K. Price: *Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces.* Journal of Global Optimization. **11** (1997), 341–359.
6. M. Georgioudakis and V. Plevris: *A Comparative Study of Differential Evolution Variants in Constrained Structural Optimization.* Front. Built Environ. **6** (2020), 102.
7. J. Ma and H. Li: *Research on Rosenbrock Function Optimization Problem Based on Improved Differential Evolution Algorithm.* Journal of Computer and Communications. **7** (2019), 107-120.
8. P. Yu. Gubin, V. P. Oboskalov, A. Mahņitko and A. Gavrilovs: *An Investigation into the Effectiveness of the Differential Evolution Method for Optimal Generating*

Units Maintenance by EENS Criteria. In: 61th International Scientific Conference on Power and Electrical Engineering (RTUCON), Riga Technical University, 2020, pp.1-5.

9. P. Yu. Gubin, V. P. Oboskalov, A. Mahņitko and R. Petrichenko: *Simulated Annealing, Differential Evolution and Directed Search Methods for Generator Maintenance Scheduling.* Energies. **13** (2020), 5381. DOI: https://doi.org/10.3390/en13205381.
10. V. Yu. Ilyichev and E. A. Yurik: *Development of methodology for calculation of optimal distribution of electric power between power units of condensing power plant.* Izvestiya MGTU MAMI. **15 (2)** (2021), 18-25.
11. W. Yang, E. M. D. Siriwardane, R. Dong, Y. Li and J. Hu: *Crystal structure prediction of materials with high symmetry using differential evolution.* J. Phys. Condens. Matter. **33** (2021), 455902.
12. C. Y. Lee and C. H. Hung: *Feature Ranking and Differential Evolution for Feature Selection in Brushless DC Motor Fault Diagnosis.* Symmetry **13** (2021), 1291.
13. S. Chakraborty, A. K. Saha, S. Sharma et al.: *Comparative Performance Analysis of Differential Evolution Variants on Engineering Design Problems.* J. Bionic. Eng. **19** (2022), 1140–1160.
14. Z. Wu, N. Cui, L. Zhao, L. Han, X. Hu, H. Cai, D. Gong, L. Xing, X. Chen and B. Zhu: *Estimation of maize evapotranspiration in semi-humid regions of Northern China Using Penman-Monteith model and segmentally optimized Jarvis model.* J. Hydrol. **22** (2022), 127483.
15. D. A. Erokhin and Sh. A. Akhmedova: *The development and investigation of the efficiency of the differential evolution algorithm for solving multi-objective optimization problems.* Siberian Journal of Science and Technology. **2** (2019), 134–143.
16. V. Charilogis, I. G. Tsoulos, A. Tzallas and E. Karvounis: *Modifications for the Differential Evolution Algorithm.* Symmetry **14** (2022), 447. https://doi.org/10.3390/sym14030447.
17. S. Prabha and R. Yadav: *Differential evolution with biological-based mutation operator.* Engineering Science and Technology **23(2)** (2020), 253-263.
18. M.-T. Su, C.-H. Chen, C.-J. Lin and C.-T. Lin: *A Rule-Based Symbiotic Modified Differential Evolution for SelfOrganizing Neuro-Fuzzy Systems.* Applied Soft Computing. **11** (2011), 4847–4858.
19. M. Eftekhari, S. D. Katebi, M. Karimi and A. H. Jahanmiri: *Eliciting Transparent Fuzzy Model Using Differential Evolution.* Applied Soft Computing. **2** (2008), 466–476.
20. M. Marinaki, Y. Marinakis and G. E. Stavroulakis: *Fuzzy Control Optimized by a Multi-Objective Differential Evolution Algorithm for Vibration Suppression of Smart Structures.* Computers & Structures. **147** (2015), 126–137.
21. C. Cotta and F. Neri: *Memetic Algorithms in Continuous Optimization.* In: Neri, F., Cotta, C., Moscato, P. (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, **vol 379** (2012). Springer, Cham.
22. P. Moscato: *On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms.* Technical Report C3P 826, California Institute of Technology, Pasadena, CA, 1989.
23. D. Rutkowska, M. Piliński and L. Rutkowski: *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte.* Wydawnictwo Naukowe PWN, Warszawa, 1999.

24. M. Wozniak: *Hybrid Classifiers: Methods of Data, Knowledge, and Classifier Combination.* Springer Publishing Company, 2013.
25. F. Valdez, J. C. Vazquez and P. Melin: *A New Hybrid Method Based on ACO and PSO with Fuzzy Dynamic Parameter Adaptation for Modular Neural Networks Optimization.* In: Castillo, O., Melin, P. (eds) Fuzzy Logic Hybrid Extensions of Neural and Optimization Algorithms: Theory and Applications. Studies in Computational Intelligence, **vol 940** (2021). Springer, Cham.
26. V. Krutikov, S. Gutova, E. Tovbis, L. Kazakovtsev and E. Semenkin: *Relaxation Subgradient Algorithms with Machine Learning Procedures.* Mathematics. **10** (2022), 3959. DOI:10.3390/math10213959.
27. V. Krutikov, V. Meshechkin, E. Kagan and L. Kazakovtsev: *Machine Learning Algorithms of Relaxation Subgradient Method with Space Extension.* In: Mathematical Optimization Theory and Operations Research: 20th International Conference, MOTOR 2021, July 2021, 477–492. https://doi.org/10.1007/978-3-030-77876-7_32
28. N. Z. Shor: *Minimization Methods for Nondifferentiable Functions.* Springer: Berlin, Germany, 1985.
29. Y. Nesterov: *Subgradient optimization.* John Wiley and Sons, Inc, 2009.
30. I. V. Sergienko and P. I. Statsyuk: *On N. Z. Shor's three scientific ideas.* Cybernetics and System Analysis. **48** (2012), 2-16.
31. V. N. Krutikov and D. V. Aryshev: *Implementation of algorithms for a peer-to-peer family of subgradient methods.* Investigated in Russia. **43** (2004), 464-473.
32. V. N. Krutikov and Ya. N. Vershinin: *Subgradient minimization method with correction of descent vectors based on pairs of learning relations.* Bulletin of the Kemerovo State University. **1(57)** (2014), 46-54. https://doi.org/10.21603/2078-8975-2014-1-46-54
33. V. N. Krutikov and N. S. Samoylenko: *On the rate of convergence of the subgradient method with a change in the metric and its application in schemes of neural network approximations.* Bulletin of the Tomsk State University. Mathematics and mechanics. **55** (2018), 22 - 37. DOI: 10/17223/19988621/55/3.
34. J. D. Lee, M. Simchowitz, M. I. Jordan and B. Recht: *Gradient descent only converges to minimizers.* In: Proceedings of the 29th Annual Conference on Learning Theory (COLT). USA, June 23-26, 2016, pp. 1246–1257.
35. Y. Drori: *The exact information-based complexity of smooth convex minimization* J. Complexity, 2017, **39**, pp. 1–16.
36. D. Kim and J. A. Fessler: *Optimized first-order methods for smooth convex optimization* Math. Prog. Ser. A (2016) V. 159(1–2), pp. 81–107.
37. V. F. Demyanov : *Nonsmooth Optimization.* In Nonlinear Optimization; Lecture Notes in Mathematics; Di Pillo, G., Schoen, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 1989, pp. 55–163.
38. V. Feoktistov and S. Janaqi: *Hybridization of differential evolution with leastsquare support vector machines.* In: Proceedings of the Annual Machine Learning Conference of Belgium and The Netherlands. BENELEARN 2004, Vrije Universiteit Brussels, Belgium, January 2004, pp. 53–57.
39. A. V. Eremeev and J. V. Kovalenko : *Optimal Recombination in Genetic Algorithms for Combinatorial Optimization Problems - Part I.* Yugoslav Journal of Operations Research. **24** (2014), 1-20.DOI: 10.2298/YJOR130731040E.

40. E. Wirsansky: *Hands-On Genetic Algorithms with Python.* Birmingham – Mumbai, Packt Publishing, 2020.

41. R. Rockafellar : *Convex analysis.* Princeton, Princeton University Press, 1997.

42. X.-S. Yang: *Nature-Inspired Optimization Algorithms: Challenges and Open Problems*. Journal of Computational Science **46** (2020), 101104.

43. M. Argáez, L. Velázquez, C. Quintero, H. Klie and M. F. Wheeler: *A Hybrid Algorithm for Global Optimization Problems*. Reliab. Comput. **15** (2011), 230-241.

44. M. Dehghani and P. Trojovský: *Hybrid leader based optimization: a new stochastic optimization algorithm for solving optimization applications*. Sci. Rep. **12** (2022), 5549. DOI: https://doi.org/10.1038/s41598-022-09514-0.

45. P. T. H. Nguyen and D. Sudholt : *Memetic algorithms outperform evolutionary algorithms in multimodal optimisation.* Artificial Intelligence. **287** (2020), 103345.