The Method of Fractal Structuring as on Evolutionary Method of Global Optimization

Maryna Antonevych, Vitaliy Snytyuk and Natalia Tmienova

Taras Shevchenko National University of Kyiv, 24 B. Havrylyshyna Str., Kyiv, 04116, Ukraine

Abstract

Decision making processes in the modern world are based on the solutions of optimization problems. The variety of such problems, the corresponding objective functions and areas for finding optimal solutions is the reason for the development of new and improvement of the known optimization methods. This paper proposes a new method of fractal structuring, which is an evolutionary method from the category of soft computing methods. A feature of this method is a quick and in-depth study of the area in which the local extremum is located and the global optimum can also be found. A fractal structuring method has been developed for finding the optimum for one-dimensional, two-dimensional and n-dimensional objective functions. The first experiments were carried out, which prove the prospects and effectiveness of this method, and also indicate the possibility of its improvement.

Keywords [1]

Optimization problem, function, evolutionary method, method of fractal structuring.

1. Introduction

A large number of modern practical problems belong to the class of constraint satisfaction problems [1]. The target functions in such tasks are, as a rule, undifferentiated and (or) poly-extremal dependencies. The use of classical methods of continuous optimization and in many cases discrete optimization is impossible [2]. Combinatorial optimization methods, evolutionary algorithms, etc - are used to solve such tasks. The functional dependencies can be set tabularly or algorithmically. In this case evolutionary algorithms are most often preferred.

Exactly, the use of this algorithms does not require strict target functions constraints, but does not guarantee the finding of a global optimum, although according to certain conditions there is a probability convergence. The obtained solutions are considered suboptimal.

Historically, the first methods of evolutionary optimization were genetic algorithms and evolutionary strategies [3, 4]. These methods allowed to consider optimization problems differently and expanded the subject base of optimization technologies. Also, these methods were based on the ideas of natural evolution. In particular, genetic algorithms traditionally use the principle that the best parents tend to have better children. Two parental potential solutions are involved in generating potential solutions-offspring. In evolutionary strategies, potential offspring solutions are generated around a single parent solution. This is the main difference between genetic algorithms and evolutionary strategies in the generation of offspring solutions.

Our hypothesis is that more potential parent solutions that will be involved in generating potential successive solutions will improve the accuracy of the solutions and speed up the convergence of the optimum search algorithms. This progress will be achieved by in-depth study of potential promising solutions and by reducing the number of algorithm steps in non-perspective directions.

2. A brief description of modern applications of evolutionary technologies

John Holland, used the ideas of genetic algorithm to study and optimize the game with one-armed and two-armed bandits (slot machines). H.-P. Schwefel and Ingo Rechenberg tried to get the part with

Information Technology and Implementation (IT&I-2021), December 01–03, 2021, Kyiv, Ukraine
EMAIL: marina.antonevich@gmail.com (A. 1); snytyuk@knu.ua (A. 2); tmyenovox@gmail.com (A. 3);
ORCID: 0000-0003-3640-7630 (A. 1); 0000-0002-9954-8767 (A. 2); 0000-0003-1088-9547 (A. 3);

the least resistance in the wind tunnel. The first results obtained using evolutionary methods testified to the prospects of this area. And the No Free Lunch Theorem [5] became the theoretical basis for the development of a set of optimization methods, each of which showed the best results in solving certain problems with a certain structure of the source data.

Modern technologies of evolutionary modeling, as a rule, are focused on the further development of the theory of evolutionary optimization and its practical application. In the first direction, we will pay attention to only a few known results. In particular, in the field of evolutionary algorithms is the famous school of Kalyanmoy Deb, a famous Indian professor. Recent studies of this school are aimed at solving multicriteria optimization problems using additional options that allow you to solve relevant problems more accurately and quickly [6, 7]. An excellent overview of multicriteria optimization methods using such options in evolutionary algorithms, in particular with the choice of informative factors and data normalization methods, is proposed in [8, 9].

Hybrid evolutionary methods, which combine technologies of fuzzy set theory, particle swarm optimization and genetic algorithms, are proposed in [10]. Two more works are devoted to the development of new methods of evolutionary optimization: the evolutionary method based on centers of mass [11] and the method of deformed stars [12]. The latter method is based on the hypothesis of involving more potential parent solutions in the generation of potential descendant solutions, which makes the search for the global optimum more informative, as well as a deeper study of the perspectives of promising potential solutions.

Thus, the results of this brief review indicate the continued development of the theory of evolutionary optimization and practical applications of evolutionary methods.

3. The method of fractal structuring in one-dimensional case

The known optimization problem in one-dimensional case can be mathematically formulated as follows:

$$\text{maximize } f(x)$$

$$\text{subject to } x \in D \subseteq R \tag{1}$$

where x is a solution in the feasible region D, and D is some segment $[a,b]$.

There are no restrictions on the function $f(x)$, in the general case. The function $f(x)$ can be set analytically, tabularly or algorithmically. The proposed method contains such steps.

Step 1. Initialize the method parameters: $n, t = 0, m_i = 7, i = \overline{1,n}$. { n is the number of potential parenting solutions in the population, t is the iteration number, m_i is the number of solutions-offsprings of the i-th parent solution}.

Step 2. Generate n uniform distributed potential solutions x_i of problem (1) on the segment $[a,b], i = \overline{1,n}$ (population P_t).

Step 3. For each solution x_i, we find the value $f(x_i), i = \overline{1,n}$.

Step 4. Create solutions-offsprings $x_i^{j_i}$, $j_i = \overline{1,m_i}, i = \overline{1,n}, x_i^{j_i} = x_i + \xi(N(0,\delta_i))$ for each x_i, where $\xi(N(0,\sigma_i))$ is the normally distributed random variable with mean 0 and standard deviation σ_i.

Step 4.1. $s_L = 0, s_R = 0, m_L = 0, m_R = 0.$

Step 4.2. For each $j_i = \overline{1,m_i}$:

Step 4.2.1. If $x_i^{j_i} < x_i$, then $\{s_L = s_L + x_i^{j_i}, m_L = m_L + 1\}$, otherwise $\{s_R = s_R + x_i^{j_i}, m_R = m_R + 1\}$.

Step 4.3. $x_L^* = \frac{1}{m_L} s_L, x_R^* = \frac{1}{m_R} s_R.$

Step 4.4. If $f(x_L^*) > f(x_R^*)$, then $x_i^H = x_L^*$, else $x_i^H = x_R^*$.

Step 4.5. Write x_i^H to the temporary population of an solutions-offsprings P_t^{test}.

Step 5. Write the elements P_t and P_t^{test} to the population P_t^{in}, find the values of the function f for the elements of the population P_t^{test}, arrange the elements of the population P_t^{in} in descending order of the values of the function f and determine the n prospective potential solutions.

Step 6. If the stop condition is not fulfilled, the iterative process continues (return to the step 3). If the stop condition is satisfied, then the value of the potential solution, which corresponds to the maximum value of the function will be the solution of the problem (1).

Traditionally, in a classic evolutionary strategy, each potential parental solution generates the same number of potential solutions, no matter how promising that solution may be. Next, a new procedure will be proposed, according to which not all parent solutions will be able to generate child solutions. The number of descendant solutions in all parental solutions will be different.

4. The method of fractal structuring in two-dimensional case

In two-dimensional case the known search problem is considered

$$\begin{gathered}\text{maximize } f(x_1,x_2)\\ \text{subject to } x=(x_1,x_2)\in D\subseteq R^2\end{gathered} \tag{2}$$

where x is a solution in the feasible region D, and D is some rectangle

$$\begin{aligned}&D=\{(x_1,x_2)\,|\,x_1\in[p_1;p_2],x_2\in[q_1;q_2]\},\\&p_1,p_2,q_1,q_2\in R.\end{aligned}$$

The properties of the function f are the same as in the previous one-dimensional case. The corresponding method will contain steps like this.

Step 1. Initialize the parameters $L=1,\ T=T_{\max},\ t=0,\ n.$ L is a parameter of the method, T in similar methods plays the role of temperature and is initially equal to a large number $T_{\max}$, t is the iteration number, n is the number of potential parenting decisions in the population.

Step 2. Generate n points-solutions $P_t=\{(x_1^1,x_2^1),(x_1^2,x_2^2),...,(x_1^n,x_2^n)\}$ uniformly distributed in D.

Step 2.1. Find the values of the function f at points P_t and obtain $f_1,f_2,...,f_n$.

Step 3. Plot virtual circles with centers at the points P_t and radiuses $r_1,r_2,...,r_n$. We will require that the circles will be placed completely in D. All radiuses must initially be considered as equal to each other, $r=\min\{(p_2-p_1),(q_2-q_1)\}/n.$

Step 4. Let $L=\dfrac{L}{t+1}$. For each i^{th} point from P_t we generate 7 solutions-offsprings. If (x_1^i,x_2^i) - potential parent solution, coordinates of the solution-offspring are

$$\begin{aligned}&x_1^{H_k}=random(x_1^i-3Lr_i;x_1^i+3Lr_i),\\&x_2^{H_k}=x_2^i+\alpha\sqrt{r_i^2-(x_1^{H_k}-x_1^i)^2},\\&\alpha=random\{-1;1\},\ k=\overline{1,7},i=\overline{1,n}.\end{aligned}$$

Parental solutions and solutions-offsprings with coordinates $(x_1^{H_k},x_2^{H_k}),k=\overline{1,7}$ are recorded to the population P_v.

The population P_v consists of $7n+n=8n$ potential solutions (fig.1). Arrange them in descending order of the objective function values.

Step 4 aims to explore the area around the potential solution of the optimization problem. This approach does not protect us from hitting the local extreme. To prevent this, we suggest the following steps:

Step 5. Select $2n$ best (upper) potential solutions from the population P_v. Let's form $2n$ pairs, $i, j = random\{1,2,...,m\}, i \neq j$ and get $2n$ new potential solutions:

$$x_1^l = \left(\frac{x_1^i + x_1^j}{2}, \frac{x_2^i + x_2^j}{2}\right), \; l = \overline{1,2n}.$$

If a pair of promising solutions are close together, their average value will allow you to explore more deeper the area around these solutions (it is assumed that they are at short distances from each other) and possibly find a better solution.

If the solutions are far from each other, then finding their average value is an attempt to expand the search area and, as in the previous case, it is possible to find a better solution.

Step 6. Select the $2n$ worst solutions from the population P_v. We find also the average value of the objective function f_{ave} for the n best solutions. For each of the worst solutions (x_1^l, x_2^l), take the following steps:

Step 6.1. Give a small increment $(\Delta x_1^l, \Delta x_2^l)$, generated accordingly by a uniform distribution, where:

$$\Delta x_1^l \in \left(x_1^l - \frac{p_2 - p_1}{n};\; x_1^l + \frac{p_2 - p_1}{n}\right),$$
$$\Delta x_2^l \in \left(x_2^l - \frac{q_2 - q_1}{n};\; x_2^l + \frac{q_2 - q_1}{n}\right), \; l = \overline{1,2n}.$$

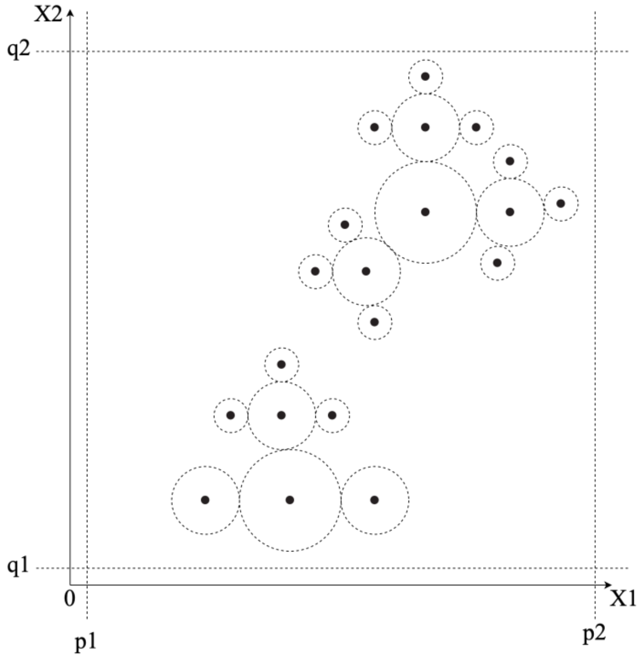

Figure 1: Fractal structure of potential solutions

If $f(x_1^l + \Delta x_1^l, x_2^l + \Delta x_2^l) > f_{ave}$, so $(x_1^H, x_2^H) = (x_1^l + \Delta x_1^l, x_2^l + \Delta x_2^l)$ - is a new potential solution and it is recorded to the population P_w. Otherwise, take a random number $r \in (0,1)$ and if

$$r < P(\min(\Delta x_1^l, \Delta x_2^l)) = \exp(-\min(\Delta x_1^l, \Delta x_2^l) / T),$$

then (x_1^H, x_2^H) is recorded to the population P_w.

If $r > P(\min(\Delta x_1^l, \Delta x_2^l))$, we move on to the next solution from the set of the worst.

If the set of the worst solutions is exhausted and the stop criterion is not met, then $T = \frac{T}{2}$, $t = t + 1$ and we go to the step 4. If the stop criterion is met, then it is the end of the algorithm. Thus, the population of the new epoch is formed from better solutions obtained from $2n$ elements of the population P_w; solutions ($2n$) calculated in step 5 and solutions ($8n$) from the population P_v, so that the total number of them is equal to n.

5. The method of fractal structuring in n-dimensional case

In the n-dimensional case, the optimization problem

$$\text{maximize } f(x_1, x_2, ..., x_m) \tag{3}$$

is considered, where x is a solution in the feasible region D,

$$x = (x_1, x_2, ..., x_m) \in D \subseteq R^m$$

and D is some rectangular hyperparallelepiped

$$D = \{(x_1, x_2, ..., x_m) \mid x_i \in [a_i, b_i], i = \overline{1,m}\},\ a_i, b_i \in R, i = \overline{1,m}.$$

In some cases, data normalization is applied and the area D is a hypercube

$$D = \{(x_1, x_2, ..., x_m) \mid x_i \in [0,1], i = \overline{1,m}\}.$$

The following algorithm for solving problem (3) is proposed.

Step 1. Perform the initialization of the algorithm parameters. Iteration number (population of potential solutions) is $t = 1$.

Step 2. Generate a sample of uniform distributed in the hypercube points

$$P_t = \{(a_1^1, a_2^1, ..., a_m^1), ..., (a_1^n, a_2^n, ..., a_m^n)\},\ a_i^j \in (0,1), \forall i = \overline{1,m}, \forall j = \overline{1,n}.$$

Step 2.1. Find the value of the function f at the points in the sample P_t and get $f_1, f_2, ..., f_n$.

Step 3. Assume that each point in the sample P_t is the center of hypersphere with radius $r = \frac{1}{n}$. We will require that each point of the hypersphere lie completely inside the hypercube $[0,1]^n$.

The equation of such hyperspheres:

$$\sum_{i=1}^{m} (x_i - a_i^j)^2 = r^2,\ j = \overline{1,n}.$$

Step 4. For each j-th hypersphere we generate 7 offsprings solutions (points) that will lie on its surface and are the centers of the hyperspheres with radius $r = \frac{r}{t+1}$. To find such a point we generate a uniformly distributed random number $k = random\{1, 2, .., m\}$. Next we generate a random vector $(x_1, x_2, ..., x_{k-1}, x_{k+1}, ..., x_m)$, such that

$$(x_i \in (a_i^j - r, a_i^j + r), i \neq k) \text{ або } \Big(\sum_{\substack{i=1 \\ i \neq k}}^{m} (x_i - a_i^j)^2 < r^2\Big),$$

and calculate

$$x_k = a_k^j + random\{-1;1\} \cdot (r^2 - \sum_{\substack{i=1 \\ i \neq k}}^{m} (x_i - a_i^j)^2).$$

We obtain a point with coordinates $(x_1, x_2, ..., x_m)$, lying on the parent hypersphere. Let's recognize its elements $b_i^{jl} = x_i$, where i determines the coordinate, j is the number of the parent solution, l is the number of the offspring-solution, $l = \overline{1,7}$. We generate such points for each potential parental solution and record all parental and offspring solutions in the population P_v. The number of such a population elements will be $8n$.

The next steps of the algorithm will be to ensure the "diversity" of the population of potential solutions and to avoid hitting the local optimums. Next, we propose data transformations that will play the role of mutations, as well as focus on a more detailed study of promising areas and a random search in a wide range of unpromising solutions.

Step 5. If a pair of prospective solutions are close together, researching their averages and values around them will allow you to explore the prospective area more deeply (assuming they are at a short distance from each other) and possibly find a better solution. If the solutions are far from each other, then finding their average value is an attempt to expand the search area and, as in the previous case, it is possible to find a better solution. Let $a = (a_1, a_2, ..., a_m)$ and $b = (b_1, b_2, ..., b_m)$ are promising potential parenting solutions. Then the point that lies in the middle of the segment connecting the points a and b has the following coordinates

$$c = \left(\frac{a_1 + b_1}{2}, \frac{a_2 + b_2}{2}, ..., \frac{a_m + b_m}{2}\right).$$

Suppose, too, that the best offspring solutions may lie in the middle of the rectangle sides for which the segment connecting the points a i b is a diagonal. To generate them, we generate a random number $r \in \{-1, 1\}$, where (-1) corresponds to the solution a, 1 corresponds the solution b and the random number $q \in \{1, 2, ..., m\}$. Generate the descendant vector as follows:

$$c = (\chi(r = -1)a_1 + \chi(r = 1)b_1, ..., \frac{a_q + b_q}{2}, ..., \chi(r = -1)a_m + \chi(r = 1)b_m).$$

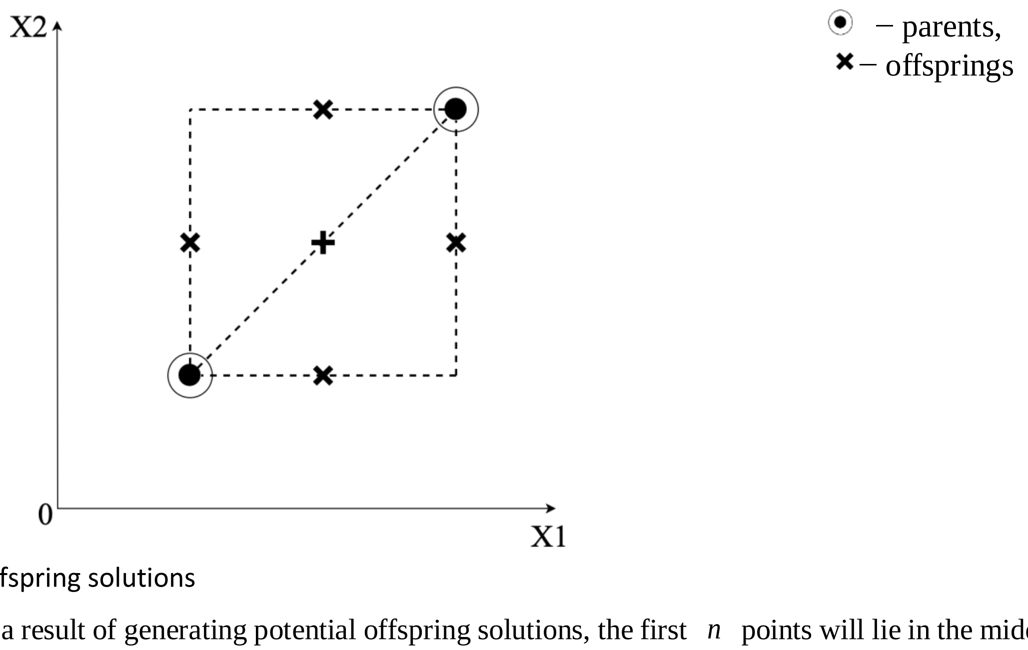

Figure 2: Offspring solutions

Thus, as a result of generating potential offspring solutions, the first n points will lie in the middle of the main diagonal of the hypercube (rectangular hyperparallelepiped) (the ends of this diagonal are the parent potential solutions), the other points will be in the middle of the side edges of such hypercube or hyperparallelepiped. This way of generating potential offspring solutions allows us to explore the area between the best solutions, as well as to test the hypothesis that parental solutions may be improved by changing one of the coordinates.

Step 6. Just as there may be an even better solution between the best potential solutions, so, given the relief of many polyextreme functions, the best solution may be among the worst potential solutions. So, like the two-dimensional case,
let's choose $2n$ the worst solutions from the set P_v.
Similarly, we find the average value of the objective function f_{ave} for the n best solutions. For each of the worst potential solutions $(x_1^l, x_2^l, ..., x_m^l)$, follow these steps.

Step 6.1. Let's give a small random increment

$$(\Delta x_1^l, \Delta x_2^l, ..., \Delta x_m^l),$$

where

$$\Delta x_i^l \in (x_i^l - \frac{1}{n}, x_i^l + \frac{1}{n}), \text{ if } D = \{(x_1, x_2, ..., x_m) \mid x_i \in [0,1], i = \overline{1,m}\}$$

and

$$\Delta x_i^l \in (x_i^l - \frac{b_i - a_i}{n}, x_i^l + \frac{b_i - a_i}{n}), \text{ if } D = \{(x_1, x_2, ..., x_m) \mid x_i \in [a_i, b_i], i = \overline{1,m}\},\ a_i, b_i \in R,\ i = \overline{1,m},$$

$$l = \overline{1,2n}.$$

It is also possible when a random number $p = random\{1, 2, ..., m\}$ is generated and a random increment of only one coordinate is provided:

$$\Delta x_p^l \in (x_p^l - \frac{1}{n}, x_p^l + \frac{1}{n}), \text{ або } \Delta x_p^l \in (x_p^l - \frac{b_p - a_p}{n}, x_p^l + \frac{b_p - a_p}{n}).$$

Other coordinates of potential solutions remain unchanged, ie

$$\Delta x_q^l = 0\ \forall q = \overline{1,m},\ q \neq p.$$

So, if $f(x_1^l + \Delta x_1^l, x_2^l + \Delta x_2^l, ..., x_m^l + \Delta x_m^l) > f_{ave}$, then

$$(x_1^H, x_2^H, ..., x_m^H) = (x_1^l + \Delta x_1^l, x_2^l + \Delta x_2^l, ..., x_m^l + \Delta x_m^l)$$

Is a new potential solution that is being recorded in the population P_w. Otherwise, generate a uniformly distributed random number $r \in (0,1)$ and if

$$r < P(\min(\Delta x_1^l, \Delta x_2^l, ..., \Delta x_m^l)) = \exp(-\min(\Delta x_1^l, \Delta x_2^l, ..., \Delta x_m^l) / T),$$

then $(x_1^H, x_2^H, ..., x_m^H) = (x_1^l + \Delta x_1^l, x_2^l + \Delta x_2^l, ..., x_m^l + \Delta x_m^l)$ write to the new population P_w. If so $r \geq P(\min(\Delta x_1^l, \Delta x_2^l, ..., \Delta x_m^l))$, let's move on to the next from $2n$ worst solutions.
If the set of the worst solutions have been exhausted and the stop condition is not met, then reduce the temperature $T = \frac{T}{2}$, $t = t + 1$ and go to step 4.

If the stop condition is satisfied, then we have the end of the algorithm.
Thus, the population of the new iteration is formed from the best solutions obtained from $8n$ population P_v solutions, $2n$ solutions from population P_w and n solutions obtained in step 5.

6. Algorithm for improving the process of formation of the offspting solutions population

According to the conditions of the implementation of the evolutionary strategy [13-15], we initially assume that each parent potential solution can have the same number of offspring solutions, which is the reason for the long-term convergence of the algorithm. In order to speed it up, we propose to use the following hypotheses.

Hypothesis 1. It is more probably that a better offspring solution lies around a better parental solution than a worse one.

Hypothesis 2. To find a better offspring solution, it is rational to generate more offspring solutions in a neighborhood of better parent solution than in a worse one.

We will propose an appropriate procedure for generating offspring solutions and verify it at the end of the study.

Suppose there are n parental solutions, we need to get all the $7n$ potential solutions for posterity. Let the set of potential solutions contain the following elements:

$$x^l = (x_1^l, x_2^l, ..., x_m^l),\ l = \overline{1,n}\ .$$

Find the values of the objective function at points:

$$f^l = f(x^l) = f(x_1^l, x_2^l, ..., x_m^l),\ l = \overline{1,n}.$$

Arrange the sequence $\{x^l\}_{l=1}^n$ in descending order of values f^l. Let's divide this sequence into one of the ratios (50:50, or 60:40, or 70:30), where the first number means the percentage of the best solutions, the second number is the percentage of the worst solutions that will be removed. Let's take appropriate action. Let the number of remaining solutions z.

Let's perform normalization of values $f^l,\ l = \overline{1,n}$:

Then the l-th parental solution x^l will have $N^l = [\overline{f}^l \cdot 7n]$ descendant solutions, $l = \overline{1,z}$.

The magnitude of the parent potential solution neighborhood in which potential offspring solutions will be generated is determined by the researcher and depends on the standard deviation value.

The following hypothesis is to be studied.

Hypothesis 3. Over time, the value of standard deviation used to generate potential offspring solutions for the best parents should decrease, and for the worst parents should increase.

The realization of this hypothesis will be aimed at finding the best solution in the neighborhood of the best parent solutions, and the large value of standard deviation will play the role of mutations in evolutionary algorithms and allow us to explore a wider area with the prospect of finding a global optimum [16].

7. Experimental verification of the obtained results

The developed new method of fractal structuring requires a large number of experiments that would confirm its effectiveness. In this paper, we present the results of only one of the simplest experiments for the one-dimensional optimization problem. However, further experiments for more complex cases also confirm the viability of the method.

Consider the problem

$$\text{maximize}\ \ f(x) = \frac{\sin(10\pi x)}{2x} + (x-1)^4,\ x \in [0,5; 2,5].$$

The graph of this function is shown in Fig. 3.

The dynamics of the objective function by iterations of the fractal structuring method is shown in Fig. 4.

Despite the fact that the stop criterion determined the maximum number of 100 iterations, the algorithm found the global maximum $f_{\max} = 5.062389479647202$ at a point $x_{\max} = 2.499984685195675$ in 9 iterations.

Experiments with other algorithms showed that the optimal value of the objective function was found by the method of deformed stars in 14 iterations, the genetic algorithm - in 50 iterations, and the method of evolutionary strategy in 50 iterations to the global optimum did not match. Such results convincingly support the fractal structuring method.

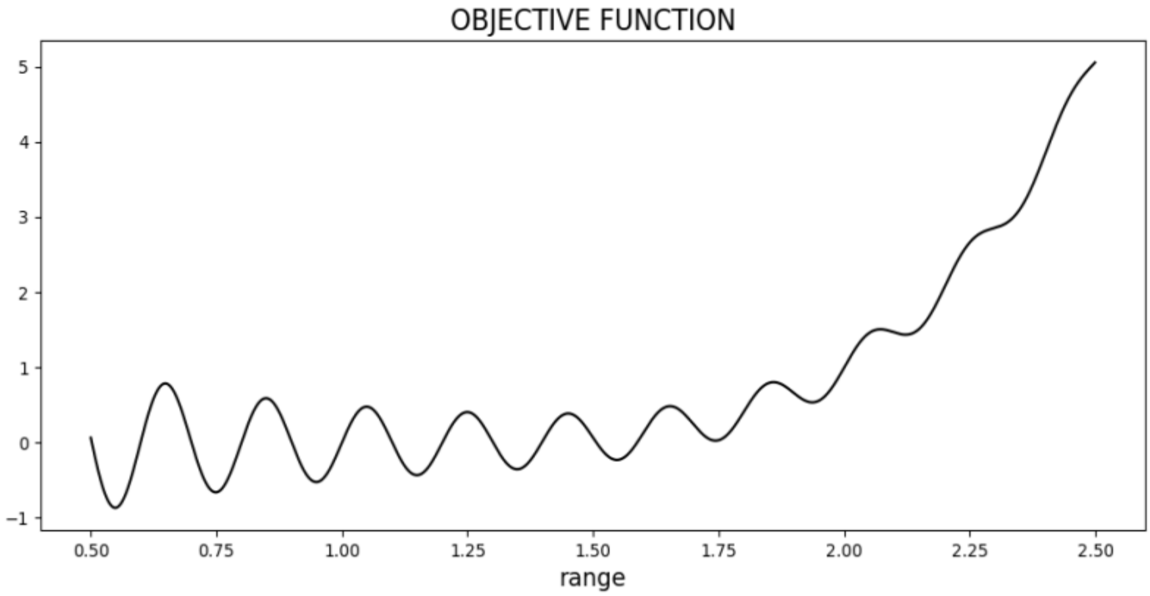

Figure 3: Objective function

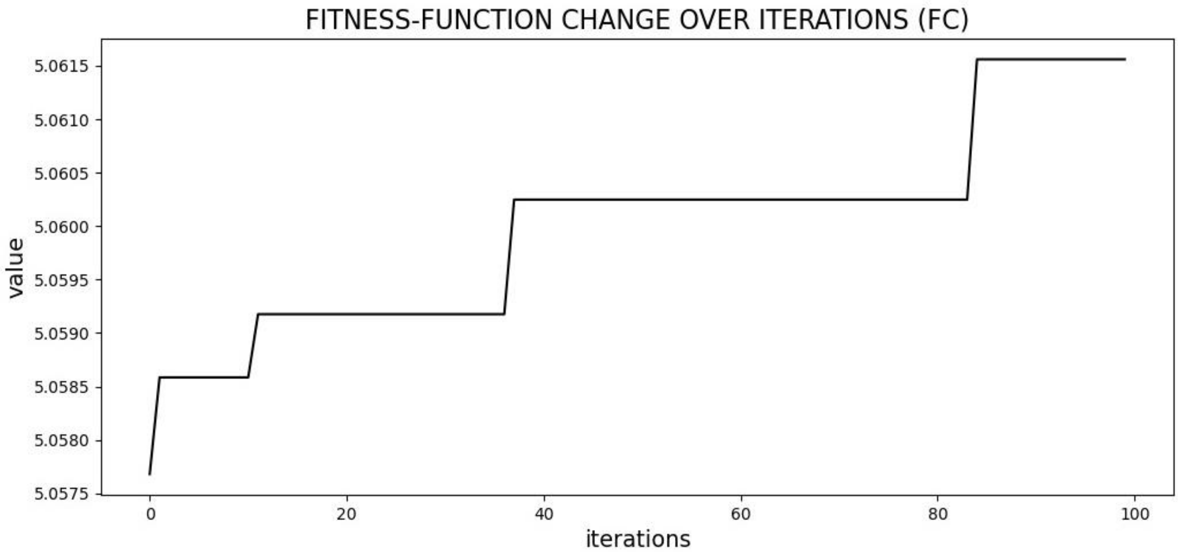

Figure 4: The dynamics of the objective function by iterations

8. Conclusion

In this paper, we propose a new method of fractal structuring. Features of its realization for a one-dimensional case, a two-dimensional case for a case of n-dimensional optimization are considered. In addition, a procedure has been developed to identify promising parental solutions and to determine the number of generated solutions in each of them. The method of fractal circles demonstrates convincing results in its effectiveness. The method is parametric and allows to search in the given region. Its main idea is a fractal search around some areas. The obtained results testify to the fast convergence of the algorithm of the fractal structuring method and considerable accuracy.

9. References

[1] H. Hnatiienko, Choice Manipulation in Multicriteria Optimization Problems, in: Proceedings of Selected Papers of the XIX International Scientific and Practical Conference "Information Technologies and Security" (ITS 2019), 2019, pp. 234–245.

[2] H. Hnatiienko, N. Tmienova, A. Kruglov, Methods for Determining the Group Ranking of Alternatives for Incomplete Expert Rankings, in: S. Shkarlet, A. Morozov, A. Palagin. (Eds.), Mathematical Modeling and Simulation of Systems (MODS'2020), MODS 2020, Advances in Intelligent Systems and Computing, Springer, Cham, volume 1265, 2021, pp. 217-226. doi:10.1007/978-3-030-58124-4_21.

[3] J. H. Holland, Genetic Algorithms, Scientific American, volume 267, no. 1, 1992, pp. 66-73.

[4] H.-P. Schwefel, Numerical Optimization of Computer Models, Wiley, Chichester, 1981.

[5] D. H. Wolpert, W. G. Macready, No Free Lunch Theorems for Optimization", IEEE transactions on evolutionary computation 1, 1997, p. 67.

[6] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, in: Proceedings of IEEE transactions on evolutionary computation, vol. 6, no. 2, 2002, pp. 182-197.

[7] K. Deb, H. Jain, An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints, in: Proceedings of IEEE Transactions on Evolutionary Computation, volume 18, no. 4, 2014, pp. 577-601, doi:10.1109/TEVC.2013.2281535.

[8] B. Xue, M. Zhang, W. N. Browne, X. Yao, A Survey on Evolutionary Computation Approaches to Feature Selection, in: Proceedings of IEEE transactions on evolutionary computation, volume 20, no. 4, 2016, pp. 606-626.

[9] L. He, H. Ishibuchi, A. Trivedi, H. Wang, Y. Nan, D. Srinivasan, A Survey of Normalization Methods in Multiobjective Evolutionary Algorithms, in: Proceedings of IEEE transactions on evolutionary computation, volume 25, no. 6, 2021, pp. 1028-1048.

[10] F. Valdez, P. Melin, O. Castillo, O. Montiel, A New Evolutionary Method with a Hybrid Approach Combining Particle Swarm Optimization and Genetic Algorithms using Fuzzy Logic for Decision Making, in: Proceedings of 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), 2008, pp. 1333-1339, doi:10.1109/CEC.2008.4630968.

[11] JA. Mejía-de-Dios, E. Mezura-Montes, A New Evolutionary Optimization Method Based on Center of Mass, in: Deep K., Jain M., Salhi S. (Eds.), Decision Science in Action, Asset Analytics (Performance and Safety Management), Springer, Singapore, 2019, doi:10.1007/978-981-13-0860-4_6.

[12] N. Tmienova, V. Snytyuk, Method of deformed stars for global optimization, in: Proceedings of the 2020 IEEE 2nd International Conference on System Analysis & Intelligent Computing (SAIC), Kyiv, Ukraine, 2020, pp. 1-4. doi:10.1109/SAIC51296.2020.9239208.

[13] A. Auger and N. Hansen, A restart CMA evolution strategy with increasing population size, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005), IEEE Press, 2005, pp. 1769-1776.

[14] P. R. Sampaio, N. Hansen, D. Brockhoff, A. Auger, A. Atamna, A methodology for building scalable test problems for continuous constrained optimization, Gaspard Monge Program for Optimisation (PGMO), ParisSaclay, 2017.

[15] R. Mallipeddi, P. N. Suganthan, Differential evolution with ensemble of constraint handling techniques for solving CEC 2010 benchmark problems, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010, Barcelona Spain, 2010, pp. 1–8. doi:10.1109/CEC.2010.5586330.

[16] S. Maniyappan, N. Umeda, A. Maki, Y. Akimoto Y, Effectiveness and mechanism of broaching-to prevention using global optimal control with evolution strategy (CMA-ES), in: Proceedings of J Mar Sci Technol, 2020, doi:10.1007/s00773-020-00743-4