Combined non-convex optimization algorithms based on differential evolution, harmony search, firefly, and L-BFGS methods

P Sorokovikov and A Gornov
Matrosov Institute for System Dynamics and Control Theory of SB RAS, 134, Lermontov Street, Irkutsk, 664033, Russian Federation

E-mail: sorokovikov.p.s@gmail.com, gornov@icc.ru

Abstract. The paper presents an approach to the numerical investigation of the problems of finding a global optimum of multiextremal functions, based on the use of bioinspired and local search methods. Three combined non-convex optimization algorithms are proposed and implemented. As globalized bioinspired methods, differential evolution, harmony search, and firefly methods are used. For local descents, we used the L-BFGS method. The numerical study of modifications of the implemented approach has been carried out using known non-convex test functions. The developed algorithms have been applied to investigate the problem of the optimal orientation of the aircraft in space. The obtained numerical results allowed us to demonstrate the efficiency of the proposed algorithms.

1. Introduction
The problem of finding the global extremum of a non-convex function continues to be one of the most difficult both in optimization theory and in practical applications. The emergence of new problems leads to the need to develop new methods for their investigation. In recent years, research on the development of numerical methods for nonlocal search has been carried out very active in many scientific organizations. To date, experts have presented a large number of methods and algorithms of various genesis [1], differing from each other in computational characteristics.

The popularity of search metaheuristic algorithms for non-convex optimization is growing every day, which is associated with the growth of computing power and the need to solve complex problems of large dimensions. These methods are inspired by nature, both animate and inanimate. They are also called “bioinspired”, “bionic”, “population”, “multi-agent”, “swarm”, “intellectual”, “behavioral”, etc. [2]. These algorithms operate with a multitude of agents at once (population) and represent an alternative to the classic “trajectory” search algorithms. Bioinspired methods can be classified as follows:

- evolutionary algorithms (genetic algorithms, differential evolution method, evolutionary strategies, etc.) [2–7];
- “wildlife-inspired” algorithms, including swarm intelligence methods (particle swarm, ant colony, bees, biogeography-based, firefly, flower pollination, cat swarm, grey wolf, cuckoo search, bacterial foraging algorithms, etc.) [2, 8–13];
• algorithms inspired by inanimate nature, human society (harmony search, teaching–learning-based optimization, cultural algorithms, mind evolution, gravitational, electromagnetic, stochastic diffusion search algorithms, etc.) [2, 8, 14–18]; others.

Any successful search strategy is based on the balance between global scanning of the feasible set and local refinement of the obtained approximations. Therefore, every method of searching for a global optimum should include two phases: a scanning exploration on the whole admissible set and an exact local search in those areas where the global extremum is most probable. This paper proposes an approach that uses the advantages of bioinspired methods for the exploration of the admissible set and gradient methods for local search. As globalized bioinspired methods, we used three algorithms of various genesis: differential evolution, harmony search, and firefly methods. For local descents, the quasi-Newton L-BFGS method [19–21] is used. The proposed approach allows the design of computational schemes that completely satisfy the above requirements.

The rest of the paper is organized as follows: Section 2 gives the formulation of the global optimization problem; Section 3 describes the proposed combined algorithms based on bioinspired and local search methods; Section 4 contains the descriptions of computational experiments on a collection of test problems and discussed results; Section 5 concludes this paper.

2. The global optimization problem

The following box-constrained problem of global optimization is considered:

\[
f(x) \rightarrow \min, \quad x \in B, \quad B = \{x \mid x = (x_1, x_2, \ldots, x_n), \alpha_i \leq x_i \leq \beta_i, \ i = 1, n\}.
\]

Here \( f(x) \) is the non-convex and smooth objective function; \( n \) is the dimension of the vector \( x \); \( \alpha, \beta \) are vectors of parallelepiped constraints.

The presented above problem can be helpful in the investigation of optimal control problems in the following statement:

\[
\frac{dx}{dt} = f(t, x, u), \quad t \in T = [t_0, t_1], \quad x(t) \in \Omega \subset \mathbb{R}^m,
\]

where \( u = u(t) \) is the control from a class of piecewise continuous vector functions with values in a set \( U : u(t) \in U \subset \mathbb{R}^k \). We assume that the domain of permissible states of the system \( \Omega \) and the function \( f(t, x, u) \) are such that the solution of the Cauchy problem exists and is unique.

It is necessary to minimize the objective functional

\[
I_0(u) = g_0(t, x, u) \rightarrow \min.
\]

We can consider this optimization of a controlled dynamical system as a search for the extremal value of the objective function on a reachable set of the corresponding system:

\[
x^* : g_0(x) \rightarrow \min, \quad x \in D,
\]

where the \( x^* \) is the point at which is achieved the minimum value of the function. The reachable set is defined as follows:

\[
D = \{x(t_i) \in \mathbb{R}^m : \dot{x} = f(t, x(t), u(t)), x(t_0) = x^0, u(t) \in U, t \in T\}.
\]

3. Description of proposed algorithms

We proposed and implemented three combined non-convex optimization algorithms based on differential evolution, harmony search, firefly, and L-BFGS methods. This section describes them.
3.1. Algorithm based on differential evolution and L-BFGS methods

Differential evolution (DE) is a popular and efficient evolutionary algorithm based on the addition of information recorded in four randomly selected agents from the population. Rainer Storn and Kenneth Price developed the approach in 1995 (see, for example, [6, 7]).

3.1.1. Algorithmic parameters

- \( m \in [4; 1000] \subset \mathbb{Z} \) – population size (number of individuals);
- \( N_{\text{LOC}} \in [0; 1000] \subset \mathbb{Z} \) – number of iterations of the L-BFGS method;
- \( F_{\text{MUT}} \in [0; 2] \subset R \) – “mutation force” coefficient (amplitude of the introduced disturbance);
- \( CR \in [0; 1] \subset R \) – mutation probability.

3.1.2. Proposed algorithm. Two variants of the DE have been implemented: the classic one, which uses only the value of the function (v.1, \( N_{\text{LOC}} = 0 \)), and the modified one, in which the optimization trajectory is built based on the L-BFGS method (v.2, \( N_{\text{LOC}} > 0 \)), starting from each point generated, according to the “differential evolution” scheme.

**Algorithm 1 (DE).**

1. Set parameters of the algorithm, select \( x^0 \), assume \( x^{\text{REC}} = x^0 \), calculate \( f^{\text{REC}} = f(x^{\text{REC}}) \), assume \( K = 0 \).
2. Generate a random starting population \( P^0 = \{ p_1 = x^0, p_2, ..., p_m \} \).
3. Perform a selection of the starting population: make \( m \) local descents, each of \( N_{\text{LOC}} \) iterations of the L-BFGS method, each begins with its individual of the starting population, get \( \{ s_1, s_2, ..., s_m \} \).
4. Refine the record, assume \( x^{\text{REC}} = \arg\min\{ s_i, l = 1, m \} \).
5. Replace in the population \( P^0 \) of the individuals \( \{ p_1, p_2, ..., p_m \} \) by the individuals \( \{ s_1, s_2, ..., s_m \} \). On the \( K \)-th iteration:
6. If the stopping criteria are met, then print \( x^{\text{REC}} \), complete the algorithm.
7. Generate a trial population. In a loop for everyone \( l = 1, m \):
   7.1. Generate a random number \( r_l \in [0, 1] \subset R \).
   7.2. If \( r_l < CR \), then assume \( p^{K+1}_l = p^K_l \). Otherwise generate three random indexes \( j_1, j_2, j_3 : j_1 \neq j_2 \neq j_3 \).
   7.3. Generate an individual of a new population \( P \supset \{ p_j = p_h + F_{\text{MUT}} \cdot (p_{j_1} - p_h) \} \).
8. Estimate all individuals: in a loop for \( l = 1, m \) perform \( N_{\text{LOC}} \) iterations of the L-BFGS method, each begins with its own individual in the population \( p_l \), get \( \{ s_1, s_2, ..., s_m \} \).
9. Refine the record, assume \( x^{\text{REC}} = \arg\min\{ s_i, l = 1, m \} \).
10. Perform selection: add \( m \) best individuals from the two populations \( P^K \) and \( \{ s_1, s_2, ..., s_m \} \) to the new population, form \( P^{K+1} \).
11. Assume \( K = K + 1 \), go to step 6.

3.2. Algorithm based on harmony search and L-BFGS methods

The harmony search (HS) algorithm was proposed in 2001 by Geem Z.W. [14]. The HS is inspired by the process of musicians seeking harmony in music. The situation of ideal harmony of sounds is...
associated with a global extremum in the problem of multidimensional optimization, and the process of musician’s improvisation is the procedure for finding this optimum (see, for example, [8, 14, 15]).

3.2.1. Algorithmic parameters

- \( m \in [4; 1000] \subseteq \mathbb{Z} \) – population size (number of harmonies);
- \( N_{LOC} \in [0; 1000] \subseteq \mathbb{Z} \) – number of iterations of the L-BFGS method;
- \( P_{HMCR} \in [0; 1] \subseteq R \) – harmony memory consideration rate;
- \( P_{PAR} \in [0; 1] \subseteq R \) – pitch adjustment rate;
- \( F_{DAMP} \in [0.5; 1] \subseteq R \) – damping factor of fret width.

3.2.2. Proposed algorithm. We implemented two variants of the HS: the basic one (v.1, \( N_{LOC} = 0 \)) and the combined with the L-BFGS method for local search (v.2, \( N_{LOC} > 0 \)).

**Algorithm 2 (HS).**

1. Set parameters of the algorithm, select \( x^0 \), assume \( x^{REC} = x^0 \), calculate \( f^{REC} = f(x^{REC}) \), assume \( K = 0 \).
2. Generate a random starting population (harmony memory) \( P^0 = \{ p_1 = x^0, p_2, \ldots, p_m \} \).
3. Perform a selection of the harmony memory: make \( m \) local descents, each of \( N_{LOC} \) iterations of the L-BFGS method, each begins with its harmony of the starting population, get \( \{ s_1, s_2, \ldots, s_m \} \).
4. Refine the record, assume \( x^{REC} = \arg\min\{ s_i, l = 1, m \} \).
5. Replace in the population \( P^0 \) of the harmonies \( \{ p_1, p_2, \ldots, p_m \} \) by the harmonies \( \{ s_1, s_2, \ldots, s_m \} \).
6. Sets the fret width \( fW = 0.02 \cdot \| \beta - \alpha \| \).
7. On the \( K \)-th iteration:
   - If the stopping criteria are met, then print \( x^{REC} \), complete the algorithm.
   - Generate a random new population (harmony set) \( Q^K = \{ q_1, q_2, \ldots, q_m \} \).
9. Modify harmony set. In a loop for everyone \( l = 1, m \) (harmony) and \( j = 1, n \) (coordinate):
   9.1. Generate a random number \( r_j \in [0,1] \subseteq R \).
   9.2. If \( r_j < P_{HMCR} \), then the coordinate of the new harmony \( q_j^i \) is replaced with the corresponding coordinate of the randomly selected harmony from the memory: \( q_j^i = p_j^i, \) \( i \in [1, m] \subseteq Z \) is a random number.
   9.3. If \( r_j < P_{PAR} \), then the coordinate of the new harmony \( q_j^i \) changes as follows: \( q_j^i = q_j^i + fW \cdot \text{gauss}(0,1) \), where \( \text{gauss}(0,1) \) is the normal distribution.
10. Estimate all harmonies: in a loop for \( l = 1, m \) perform \( N_{LOC} \) iterations of the L-BFGS method, each begins with its own harmony \( q_l \), get \( \{ s_1, s_2, \ldots, s_m \} \).
11. Refine the record, assume \( x^{REC} = \arg\min\{ s_i, l = 1, m \} \).
12. Perform selection: add \( m \) best harmonies from the two sets \( P^K \) and \( \{ s_1, s_2, \ldots, s_m \} \) to the new harmony memory, form \( P^{K+1} \).
13. Decrease value \( fW \) as follows: \( fW = fW \cdot F_{DAMP} \).
14. Assume \( K = K + 1 \), go to step 7.
3.3. Algorithm based on firefly and L-BFGS methods

The firefly algorithm (FA) was developed in 2007 at the University of Cambridge (UK) by Young X.S. (see, for example, [9–11]). The FA uses the following model of firefly behavior: all individuals can attract each other, regardless of their gender; the attractiveness of a firefly to other individuals is proportional to its brightness; less attractive fireflies move towards the more engaging individual; the brightness of a given firefly’s radiation seen by another individual decrease with increasing distance between the fireflies; if an individual does not see a firefly brighter than itself, then it moves randomly.

3.3.1. Algorithmic parameters

- \( m \in [4; 1000] \subset \mathbb{Z} \) – population size (number of fireflies);
- \( N_{\text{LOC}} \in [0; 1000] \subset \mathbb{Z} \) – number of iterations of the L-BFGS method;
- \( \gamma \in [0; 1] \subset \mathbb{R} \) – light absorption coefficient;
- \( \beta \in [0; 3] \subset \mathbb{R} \) – attractiveness coefficient;
- \( \alpha \in [0; 1] \subset \mathbb{R} \) – mutation rate;
- \( M \in [0.1; 10] \subset \mathbb{R} \) – power coefficient.

3.3.2. Proposed algorithm. Two variants of the FA have been implemented: the classic one (v.1, \( N_{\text{LOC}} = 0 \)) and the combined with the L-BFGS method for local search (v.2, \( N_{\text{LOC}} > 0 \)).

Algorithm 3 (FA).

1. Set parameters of the algorithm, select \( x^0 \), assume \( x^{REC} = x^0 \), calculate \( f^{REC} = f(x^{REC}) \), assume \( K = 0 \).
2. Generate a random starting population of fireflies \( P^0 = \{ p_1 = x^0, p_2, ..., p_m \} \) each of which has its own light intensity \( f(p_i), i = \overline{1,m} \) (objective function value).
3. Perform a selection of the starting population: make \( m \) local descents, each of \( N_{\text{LOC}} \) iterations of the L-BFGS method, each begins with its individual of the starting population, get \( \{ s_1, s_2, ..., s_m \} \).
4. Refine the record, assume \( x^{REC} = \arg \min \{ s_l, l = \overline{1,m} \} \).
5. Replace in the population \( P^0 \) of the fireflies \( \{ p_1, p_2, ..., p_m \} \) by the individuals \( \{ s_1, s_2, ..., s_m \} \).
6. On the \( K \)-th iteration:
   6.1. If the stopping criteria are met, then print \( x^{REC} \), complete the algorithm.
7. Create a new population \( Q^K = \{ q_1, q_2, ..., q_m \} \), \( f(q_i) = \infty, i = \overline{1,m} \).
8. Modify a population \( Q^K \). In a loop for everyone \( l = \overline{1,m} \) and \( j = \overline{1,m} \):
   8.1. Calculate the attractiveness of the firefly \( p_i \) to the individual \( p_j \): \( \beta_l = \beta_0 \cdot \exp(-\gamma d_{ij}^M) \), where \( d_{ij} \) is Euclidean distance between \( p_i \) and \( p_j \).
   8.2. Generate a random number \( r_{ij} \in [-1,1] \subset \mathbb{R} \).
   8.3. The firefly \( p_i \) moves towards the individual \( p_j \). Create a new individual \( p_{\text{new}} = p_i + \beta_l(p_j - p_i) + \alpha \cdot \delta \cdot r_{ij} \), where \( \delta = 0.05 \cdot \| \beta \cdot d_i \| \).
   8.4. If \( f(p_{\text{new}}) < f(q_i) \), then \( q_i = p_{\text{new}} \).
9. Estimate all fireflies: in a loop for \( l = \overline{1, m} \) perform \( N_{\text{LOC}} \) iterations of the L-BFGS method, each begins with its own firefly \( q_l \), get \( \{s_1, s_2, ..., s_n\} \).

10. Refine the record, assume \( x^{\text{REC}} = \arg\min \{s_i, l = \overline{1, m}\} \).

11. Perform selection: add \( m \) best fireflies from the two populations \( P^K \) and \( \{s_1, s_2, ..., s_n\} \) to the new population, form \( P^{K+1} \).

12. Assume \( K = K + 1 \), go to step 6.

4. Testing of proposed algorithms

The algorithms were implemented in C using uniform software standards and investigated on a collection of test problems. The sets of algorithmic parameters are presented in table 1.

| Algorithm | Parameter values |
|-----------|------------------|
| DE (v.1)  | \( m = 10, N_{\text{LOC}} = 0, F_{\text{MUT}} = 0.5, CR = 0.9 \) |
| DE (v.2)  | \( m = 10, N_{\text{LOC}} = 30, F_{\text{MUT}} = 0.5, CR = 0.9 \) |
| HS (v.1)  | \( m = 10, N_{\text{LOC}} = 0, P_{\text{HMCR}} = 0.9, P_{\text{PAR}} = 0.1, F_{\text{DAMP}} = 0.995 \) |
| HS (v.2)  | \( m = 10, N_{\text{LOC}} = 30, P_{\text{HMCR}} = 0.9, P_{\text{PAR}} = 0.1, F_{\text{DAMP}} = 0.995 \) |
| FA (v.1)  | \( m = 10, N_{\text{LOC}} = 0, \gamma = 0.5, \beta_0 = 3.0, \bar{\alpha} = 1.0, M = 10 \) |
| FA (v.2)  | \( m = 10, N_{\text{LOC}} = 30, \gamma = 0.5, \beta_0 = 3.0, \bar{\alpha} = 1.0, M = 10 \) |

A numerical comparison of algorithms with each other is of interest. We provided the same testing conditions for each algorithm. The algorithms were launched 50 times from the same starting populations uniformly distributed in the domain of definition of the function. The dimension of the test problems is 100 variables. The population size for all algorithms is \( m = 10 \). A single stopping criterion for all algorithms is exceeding 10000 calls to the objective function. Below is a description of each test problem with the presentation of its plot (surface and level lines) and the results of comparing algorithms in the form of box and whisker diagrams for this problem (figures 1–5). The vertical axis plots the mean values of the function over the entire population, obtained from 50 runs. Table 2 presents statistics on launches: mean and standard deviation.

Computational experiments were performed using a server computer with the following characteristics: \( 2 \times \) Intel Xeon E5-2680 v2 2.8 GHz (20 cores, 40 threads); 128 Gb DDR3 1866 MHz.

4.1. Rosenbrock function

The Rosenbrock problem is a non-convex function proposed by Rosenbrock H. in 1960 [22]. Finding a global optimum for this function is a non-trivial problem due to its ravine nature.

\[
 f(x) = \sum_{i=1}^{n-1} \left( 100 \cdot (x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right), \quad B = [-2.048, 2.048]^n, \quad n = 100.
\]

Global minimum point and value: \( x_1^* = 1.0, f^* = 0.0 \).

All three modifications of the algorithms based on combinations with the L-BFGS method (v.2) showed significantly better results than the basic ones (v.1). Especially the performance of the HS has increased. Among the algorithms tested, FA (v.2) and DE (v.2) are leading, producing solutions very close to the global minimum value (see figure 1, table 2).
4.2. Rastrigin function

The Rastrigin test optimization model was proposed in 1974 by the Soviet scientist Rastrigin L.A. [23]. The function is one of the most famous separable test models for global optimization algorithms.

\[
f(x) = 10 \cdot n + \sum_{i=1}^{n} \left( x_i^2 - 10 \cdot \cos(2\pi x_i) \right), \quad B = [-5.12, 5.12]^n, \quad n = 100.
\]

Global minimum point and value: \( x^* = 0.0, \; f^* = 0.0 \).

As can be seen from figure 2, table 2, among the proposed algorithms, DE (v.2) and HS (v.2) showed the best results, significantly better compared to their base versions (v.1).

Figure 1. Rosenbrock function plot, \( n = 2 \) (on the left); box and whisker plot of numerical comparison of algorithms on the Rosenbrock problem, \( n = 100 \) (on the right).

Figure 2. Rastrigin function plot, \( n = 2 \) (on the left); box and whisker plot of numerical comparison of algorithms on the Rastrigin problem, \( n = 100 \) (on the right).
4.3. Schwefel function
The Schwefel function is one of the classic multi-extremal optimization test problems (see, for example, [24]).

\[ f(x) = 418.9829 \cdot n + \sum_{i=1}^{n} (-x_i \cdot \sin(\sqrt{|x_i|})), \quad B = [-500.0, 500.0]^n, \quad n = 100. \]

Global minimum point and value: \( x^*_i = 420.9687, \quad f^* = 0.0. \)

Among the basic variants of algorithms HS (v.1) is the leader. The proposed modification of HS (v.2) improved its efficiency, showing the best computational characteristics among the six algorithms (see figure 3, table 2).

4.4. Griewank function
Another multi-extremal test problem is the Griewank function (see figure 4). It has the following structure:

\[ f(x) = 1 + \sum_{i=1}^{n} \left( \frac{x_i^2}{4000} \right) - \prod_{i=1}^{n} \cos \left( \frac{x_i}{\sqrt{i}} \right), \quad B = [-600.0, 600.0]^n, \quad n = 100. \]

Global minimum point and value: \( x^*_i = 0.0, \quad f^* = 0.0. \)

As can be seen from figure 4 and table 2, DE (v.2) and HS (v.2) are leading.

Figure 3. Schwefel function plot, \( n = 2 \) (on the left); box and whisker plot of numerical comparison of algorithms on the Schwefel problem, \( n = 100 \) (on the right).
4.5. Ackley function

The Ackley problem is often used when testing global optimization algorithms (see, for example, [24]). Objective function has a complex system of uniformly distributed local optima (figure 5).

\[ f(x) = 20 + e - 20 \cdot e^{-0.2 \sum_{i=1}^{n} x_i^2 - \frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)} \quad B = [-30.0, 30.0]^n, \quad n = 100. \]

Global minimum point and value: \( x^*_n = 0.0, \quad f^* = 0.0. \)

On this problem, DE (v.2) significantly benefits from efficiency in comparison with other implemented algorithms (see figure 5 and table 2).
Table 2. Statistics on runs of algorithms: mean and standard deviation (SD).

| Algorithm | Rosenbrock | Rastrigin | Schwefel | Griewank | Ackley |
|-----------|------------|-----------|----------|----------|--------|
|           | Mean      | SD       | Mean      | SD       | Mean   | SD   |
| DE v.1    | 609.05    | 92.981   | 280.93    | 27.043   | 2352.78 | 6218.7 |
| DE v.2    | 0.0581    | 0.0287   | 0.0494    | 0.0305   | 7812.23 | 2073.8 |
| HS v.1    | 3274.0    | 300.85   | 549.47    | 28.910   | 11894.5 | 648.87 |
| HS v.2    | 25.068    | 14.877   | 2.9548    | 1.5472   | 1937.42 | 107.81 |
| FA v.1    | 299.81    | 61.380   | 818.99    | 69.126   | 21146.8 | 942.92 |
| FA v.2    | 0.052     | 0.0309   | 265.91    | 73.013   | 3519.01 | 157.43 |

4.6. The problem of the optimal orientation of the aircraft in space

The mathematical model of the aircraft [25] is described by the following system of differential equations:

\[
\begin{align*}
\dot{x}_1 &= x_3, \\
\dot{x}_2 &= x_4, \\
\dot{x}_3 &= -x_4 + u_1 \sin u_2, \\
\dot{x}_4 &= -x_3 + u_1 \cos u_2.
\end{align*}
\]

The controls are subject to restrictions: $0 \leq u_1(t) \leq 1$, $-\pi \leq u_2(t) \leq \pi$. The problem is to transfer the system from point $x(t_0) = (10,0,0,0)$ to point $x(t_0) = (0,0,0,0)$ in the shortest possible time $t_1$, $t \in [0, t_1]$. The objective functional is $I_0(u) = t_1 \rightarrow \min$. The selection of the optimum time was performed, minimizing the discrepancy $\sum_{i=1}^{n} \left[ x_i - x_i(t_1) \right]^2 \rightarrow \min$ using the proposed algorithms.

The found value of the functional is $I_0(u) = 10.286$. The solutions obtained by each of the proposed algorithms separately coincided with each other and the best of known value. Figure 6 shows the plots of the optimal control and the corresponding phase coordinates.

Figure 6. Optimal control and corresponding trajectories.
Based on the conducted computational experiments, we can formulate that among the tested bioinspired algorithms, there is no clear leader that would show the best results on all test problems. However, it can be seen that the DE method was in the lead more often than HS and FA. In all cases considered, the combined algorithms based on the nature-inspired methods and the L-BFGS showed significant improvements over the original population methods.

5. Conclusion
The presented approach to the numerical study of the problems of finding a global optimum of multiextremal functions is based on the use of differential evolution, harmony search, firefly, and L-BFGS methods. Three combined non-convex optimization algorithms are developed.

The testing of the implemented approach was carried out using well-known multiextremal functions. All three modifications of the algorithms based on combinations with the L-BFGS method showed significant improvements over the basic ones. The proposed technique has been applied to investigate the model problem of the optimal orientation of the aircraft in space. The implemented approach allows the design of computational schemes that be the basis of efficient algorithms for solving global optimization problems.

Acknowledgments
The reported study was funded by Russian Foundation for Basic Research, project No. 19-37-90065.

References
[1] Floudas C A and Pardalos P M 2009 Encyclopedia of optimization (New York: Springer Science & Business Media)
[2] Karpenko A P 2014 Modern search optimization algorithms. Algorithms inspired by nature (Moscow: BMSTU Publishing)
[3] Ashlock D 2006 Evolutionary computation for modeling and optimization (New York: Springer-Verlag)
[4] Back T 1996 Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms (Oxford: Oxford University Press)
[5] Whitley D 1994 Statistics and computing 4(2) 65-85
[6] Price K, Storn R M and Lampinen J A 2006 Differential evolution: a practical approach to global optimization (New York: Springer Science & Business Media)
[7] Storn R and Price K 1997 Journal of global optimization 11(4) 341-59
[8] Xing B and Gao W J 2014 Innovative computational intelligence: a rough guide to 134 clever algorithms (Cham: Springer)
[9] Yang X S 2010 Nature-inspired metaheuristic algorithms (Frome: Luniver Press)
[10] Yang X S 2010 International Journal of Bio-Inspired Computation 2(2) 78-84
[11] Yang X S and He X 2013 International journal of swarm intelligence 1(1) 36-50
[12] Simon D 2008 IEEE transactions on evolutionary computation 12(6) 702-13
[13] Karaboga D and Basturk B 2007 Journal of global optimization 39(3) 459-71
[14] Geem Z W, Kim J H and Loganathan G V 2001 Simulation 76(2) 60-8
[15] Mahdavi M, Fesanghary M and Damangir E 2007 Applied mathematics and computation 188(2) 1567-79
[16] Rao R V, Savsani V J and Vakharia D P 2011 Computer-Aided Design 43(3) 303-15
[17] Rashedi E, Nezamabadi-Pour H and Saryazdi S 2009 Information sciences 179(13) 2232-48
[18] Nasuto S and Bishop M 1999 Parallel algorithms and application 14(2) 89-107
[19] Liu D C and Nocedal J 1989 Mathematical programming 45(1-3) 503-28
[20] Mokhtari A and Ribeiro A 2015 The Journal of Machine Learning Research 16(1) 3151-81
[21] Zhu C, Byrd R H, Lu P and Nocedal J 1997 ACM Transactions on Mathematical Software (TOMS) 23(4) 550-60
[22] Rosenbrock H 1960 The Computer Journal 3(3) 175-84
[23] Rastrigin L A 1974 *Extreme control systems* (Moscow: Nauka)

[24] Ding K and Tan Y 2014 *Advances in Swarm Intelligence ICSI 2014 Lecture Notes in Computer Science* **8795** (Cham: Springer) p 66-78

[25] Grachev N I and Filkov A N 1986 *Solution of optimal control problems in system DIOS* (Moscow: Computing Center of Russian Academy of Sciences)