A Meta-heuristic for Improving the Performance of an Evolutionary Optimization Algorithm Applied to the Dynamic System Identification Problem

Ivan Ryzhikov, Eugene Semenkin and Evgenii Sopov
Department of System Analysis and Operations Research, Siberian State Aerospace University, Krasnoyarskij rabochij Avenue 31, 660014, Krasnoyarsk, Russian Federation

Keywords: Restart Operator, Meta-heuristic, Algorithm Control, Dynamical System, Linear Differential Equation, Evolutionary Strategies, System Identification.

Abstract: In this paper a meta-heuristic for improving the performance of an evolutionary optimization algorithm is proposed. An evolutionary optimization algorithm is applied to the process of solving an inverse mathematical modelling problem for dynamical systems. The considered problem is related to the complex extremum seeking problem. The objective function and a method of determining a solution perform a class of optimization problems that require specific improvements of optimization algorithms. An investigation of algorithm efficiency revealed the importance of designing and implementing an operator that prevents population stagnation. The proposed meta-heuristic estimates the risk of the algorithm being stacked in a local optimum neighbourhood and it estimates whether the algorithm is close to stagnation areas. The meta-heuristic controls the algorithm and restarts the search if necessary. The current study focuses on increasing the algorithm efficiency by tuning the meta-heuristic settings. The examination shows that implementing the proposed operator sufficiently improves the algorithm performance.

1 INTRODUCTION

Many mathematical modelling problems, classification problems and decision-making problems are reducible to black-box optimization problems. Powerful tools for solving these problems are evolution-based and nature-inspired optimization algorithms, which are stochastic search heuristics. This scientific field is developing and meets a lot of applications, (Eiben et al., 2015). To successfully solve these kinds of problems, a number of different problem-oriented modifications of algorithms are required. The number of new problems grows and their features change.

Due to the no free lunch theorem (Wolpert et al., 1997), the most efficient way to improve algorithm performance when solving this class of problems is to investigate the distinct class of optimization problems, analyze its features and then design algorithm modifications on the basis of the mined knowledge. However, some of problem classes may have similar features, and thus can form another class of problems, a superset of them. For this reason it is important to develop a meta-heuristic, a set of operators to control and/or to modify the extremum seeking algorithms for the whole superset, even if the algorithms sufficiently differ in terms of features and operators.

Most of the current optimization problems are multimodal and complex. For some of the problems the probability of reaching the global extremum area is small. Algorithms tend to reach neighborhoods of local extremum points and to stack in these areas and the global optimum cannot be found in a finite number of iterations.

The failure of algorithm parameter tuning to facilitate the discernment of global optima may be due to the presence of large basins of attraction. So on the one hand, the algorithm must find an accurate solution, but on the other, there is a necessity to explore the search space. There one meets a contradiction. To resolve it, we propose a meta-heuristic that monitors the properties of the solution seeking process and restarts it if some conditions are met.

The first implementation is the restart operator that reacts if a stagnation of populations is detected. The idea of using such a restart of an evolution-

Ryzhikov, I., Semenkin, E. and Sopov, E.
A Meta-heuristic for Improving the Performance of an Evolutionary Optimization Algorithm Applied to the Dynamic System Identification Problem.
DOI: 10.5220/0006049601780185
In *Proceedings of the 8th International Joint Conference on Computational Intelligence (IJCCI 2016) - Volume 1: ECTA*, pages 178-185
ISBN: 978-989-758-201-1

based algorithm was described in studies (Dao et al., 2014), (Fukunaga, 1988) and (Beligiannis et al., 2004). In this paper we consider another criterion for deciding when to apply the restart; we estimate the speed at which the best solution fitness value changes and if it is less than a certain parameter, the algorithm restarts. In the study (Loshchilov et al., 2012) the population properties and algorithm settings take on new values at each restart. In the current paper, the values of the settings are kept the same.

The second implementation was made in order to detect whether the current population is close to the point, where it was decided that a restart would be made. For this reason, a dataset is required. The dataset saves the histories of all restarts that have been performed. If the distance between a current solution and any other solution from the dataset is less than a certain parameter, the operator will stop the search and cause a restart.

In our previous studies a dynamical system identification problem was considered, an inverse mathematical modelling problem with the solution being searched for in a symbolic form. We describe single input and single output systems and suppose that the input control function is known. We also suppose that the observations are the distorted measurements of the system output.

The identification of LDE parameters requires a certain order of equation and an initial value. The automatic and simultaneous estimation of linear differential equation coefficients, initial value and order lead to the implementation of problem-oriented algorithm modifications. The proposed approach is based on the reduction of the identification problem to an extremum problem on a real value vector field. The objective function of the reduced problem is complex and multimodal (as previous studies prove) and can be evaluated only numerically. To solve this problem, a specific evolution-based optimization tool was developed; its origin is the evolutionary strategies optimization algorithm. The proposed algorithm is modified and designed for solving the described problem; its performance is sufficiently higher than the performance of standard evolution-based algorithms.

Population search algorithms had already been applied to parameter identification problem solving and experimental results show its reliability. A genetic algorithm is applied to the parameter identification problem for ordinary differential equations (Sersic et al., 1999), but in that study the structure of the system is already known. Another study where the genetic algorithm is used to estimate the coefficients of second order LDE is (Parmar et al., 2007). In that study the order reduction problem was considered. A similar problem is considered in (Narwal et al., 2016) but the cuckoo optimization algorithm is used. Another nature-inspired optimization algorithm, partial swarm optimization, was applied to nonlinear dynamical system linearization, (Naiborhu et al., 2013).

The results of previous studies (Ryzhikov and Semenkin, 2013) allow us to conclude that the proposed modified hybrid evolutionary strategies optimization algorithm is a reliable and efficient tool for solving optimization problems of the considered class in comparison with the evolutionary strategies algorithm with particular modifications and tuned parameters. In any event, the algorithm tends to be stacked in local optimum area.

In this study the proposed algorithm is supplemented with a meta-heuristic. The performance of the algorithm with different meta-heuristic settings was examined on a set of identification problems and its performance is compared to the initial algorithm performance.

2 IDENTIFIACTION PROBLEM AND MODIFIED HYBRID EVOLUTIONARY ALGORITHM

A linear dynamic system can be determined with its parameters: coefficients and order. Let the object to be identified be described with a linear differential equation of some unknown order k. So the system output $x(t)$ is a solution of the Cauchy problem:

$$a_k \cdot x^{(k)} + a_{k-1} \cdot x^{(k-1)} + \ldots + a_0 \cdot x = b \cdot u(t), \quad x(0) = x_0 . \tag{1}$$

The aim of the inverse modelling problem solving is to make a mathematical model that fits the observation data. In this case it is some differential equation (1), the solution of which is a function that fits the data well. The function is a solution of the Cauchy problem, which requires the initial value x_0. It means that an initial value estimation is also required to search for the model in the form of an LDE.

In the case of distorted observations and/or a small sample size it is a difficult problem to estimate the coordinates of the initial value vector. In general, there could be significant errors in the estimation of initial value derivatives. Thus, it is important to develop an approach to estimating the initial value coordinates and LDE parameters simultaneously.

Let a set $\{y_i, u_i, t_i\}, i = \overline{1,s}$, be a sample, where $y_i \in R$ is the dynamic system output measurement at the time point t_i, $u_i = u(t_i)$ is a control action and s is the size of the sample. In the current investigation it is supposed that the control function $u(t)$ is known. It is assumed that the output data $y_i, i = \overline{1,s}$ is distorted by limited additive noise $\xi : E(\xi) = 0, D(\xi) < \infty$:

$$y_i = x(t_i) + \xi_i, i = \overline{1,s}, \tag{2}$$

where the $x(t)$ function is a solution of the Cauchy problem (1). In the current work, the random value ξ is distributed uniformly and $\xi \sim U(-n_l, n_l)$, where $n_l \geq 0$ is a parameter for the noise variance control.

Without a loss in generality and if it is supposed that the system is controllable, one may assume that the system is described with the following differential equation:

$$\tilde{a}_k \cdot x^{(k)} + \tilde{a}_{k-1} \cdot x^{(k-1)} + \ldots + \tilde{a}_0 \cdot x = u(t). \tag{3}$$

If the differential equation order m is known, and we seek the solution of the identification problem as the LDE, it means that we need to estimate the following parameters: coefficients $\hat{a}_m = (\hat{a}_m, \ \ldots, \ \hat{a}_1, \ \hat{a}_0)^T \in R^{m+1}$ and initial value vector $\hat{x}(0) \in R^m$.

Since there is no information about the system order, let m be the order of the LDE model, which is assumed to be limited, $m \leq M \in N$. This limitation M is supposed to take a value that is not less than the real system order.

Now the LDE is fully determined and the model output is a solution of the initial value problem:

$$\begin{gathered} \hat{x}^{(m)} + \hat{a}_m \cdot \hat{x}^{(m-1)} + \ldots + \hat{a}_1 \cdot \hat{x} = \hat{a}_0 \cdot u(t), \\ \hat{x}(0) = x_0^m. \end{gathered} \tag{4}$$

We propose performing the estimation of all the required parameters with two vectors: the equation coefficients $\hat{a} \in R^{M+1}$ and the initial value coordinates $\hat{x}_0 \in R^M$. To estimate the order of the LDE it was supposed to use a transformation $T_1(a) : R^M \to O(M) \subset N$,

$$\begin{gathered} O(M) = \{n : n \leq M, n \in N\}, \\ T_1(a) = M - n_0\big|_{\hat{a}=a}, \end{gathered} \tag{5}$$

where $n_0\big|_{\hat{a}=a}$ is defined by the following expression:

$$n_0\big|_{\hat{a}} = \underset{n<N}{\arg\max} \left(\sum_{i=0}^{n} |\hat{a}_n| = 0 \right), \tag{6}$$

which determines the order of the differential equation.

Let A be some vector space $R^{\dim_R(A)}$, so using vectors $\hat{a}$, $\hat{x}_0$ and transformation

$$\begin{gathered} T_2(a \in A, n) : A \times N \to R^{\dim_R(A)-n}, \\ T_2(a \in A, n) = v : v_i = a_i, i = \overline{1, \dim_R(A) - n}, \end{gathered} \tag{7}$$

one can define the parameters in the notation (3).

Let $\hat{x}(t)\big|_{\hat{a}=a, \hat{x}(0)=x_0}$ be a solution of the Cauchy problem (4) with parameters: $m = T_1(a)$, $\hat{a} = T_2(a, m)$ and $\hat{x}(0) = T_2(x_0, m)$.

The conceptual formulation of the inverse mathematical modelling problem involves receiving the model from the data. In other words we need to find a solution of the extremum problem: $m^* = T_1(\hat{a}^{opt})$, $\hat{a}^*_{m^*} = P(\hat{a}^{opt}, m^*)$ and $\hat{x}^*(0) = P(\hat{x}_0^{opt}, m^*)$, where

$$\begin{gathered} \hat{a}^{opt}, \hat{x}_0^{opt} = \underset{a \in R^{M+1}, \, x_0 \in R^M}{\arg\min} I(a, x_0), \\ I(a, x_0) = \sum_{i=1}^{s} \left| y_i - \hat{x}(t_i)\big|_{\hat{a}=a, \hat{x}(0)=x_0} \right|. \end{gathered} \tag{8}$$

Thus, the simultaneous estimation of all the parameters leads to the extremum problem solving of $R^{M+1} \times R^M$.

The extremum problem (8) is complex; it can be evaluated only numerically, is multimodal and as will be shown below, has a lot of local optimum points in its basins of attraction. As we need to find the real-value vectors, it is preferable to use an optimization technique that is designed for this purpose. The evolution strategy optimization algorithm was selected, whose principles are described in (Schwefel, 1995). It is shown in many different works that it is a reliable, flexible and powerful stochastic optimization tool.

For increasing the efficiency of the optimization algorithm, the proposed approach to solving the identification problem was analysed, and so the evolutionary strategies algorithm was modified.

Every alternative of problem (8) is an individual and is characterized by the value of its fitness. The fitness function of alternative $\chi \in \mathrm{X} \subset R^{2 \cdot M+1}$ is a mapping

$$f(\chi) = \left(1 + I(\chi \to \arg(I))\right)^{-1}, \tag{9}$$

where $\chi \to \arg(I) = \langle \hat{a}, \hat{x}(0) \rangle$ is a transformation of the individual's vector coordinates to the arguments of the functional (8).

In the current study the evolutionary strategy optimization algorithm was implemented with the same features and settings as those presented in the study (Ryzhikov et al., 2016).

Let the optimization problem dimension be $n_p = \dim_R \left(R^{2 \cdot M+1} \right)$. The crossover operator is determined by one of the expressions:

$$i_r = \sum_{j=1}^{n_p} w_j \cdot i_s^j \cdot \left(\sum_{j=1}^{n_p} w_j \right)^{-1}, \qquad (10)$$

$$P\left(\left(i_c\right)_j = \left(i_s^q\right)_j \right) = w_q \cdot \left(\sum_{j=1}^{n_p} w_j \right)^{-1}, \; j = \overline{1, n_p}, \qquad (11)$$

where i_c is an offspring, i_s is one of the parents, n_p is the quantity of parents, w is the weight coefficient.

Here we describe some standard and proposed methods of weight determination. For the case (10): $w_j = \left(n_p\right)^{-1}$, $w_j = f\left(i_s^j\right)$, $w_j \sim U(\min_c, 1)$, $w_j \sim U(\min_c, f(i_s^j))$; and for the case (11): $w_j = \left(n_p\right)^{-1}$, $w_j \sim U(\min_c, f(i_s^j))$. In the given expressions the variable $\min_c$ is a crossover parameter that prevents dividing by 0.

The first essential improvement of the evolutionary strategies algorithm is the hybridization, which merges the evolution optimization with a local search strategy based on the stochastic extremum seeking algorithm of the 0th order. In our study the main algorithm was coupled with the proposed coordinate-wise extremum seeking technique. The described searching operator acts after standard operators in every generation. The aim of its implementation is to improve the alternatives after the random search. It allows the populations to be directed and the accuracy of every distinct alternative to be improved. The suggested local optimization algorithm is controlled by 4 parameters: N_1^L - the number of individuals to be optimized, N_2^L - the number of genes to be improved, N_3^L - the number of steps for every gene, and $h_L \sim U\left(0, h_L^{\max}\right)$ - the optimization step value. However, not all the genes can be improved. If the alternative represents a model with the order m, than its $M - m$ genes, which correspond the initial values coordinates, should be excluded.

The second modification is the suppression of the mutation influence. It is a very important point, because the way of transforming the objective vector into a differential equation makes the problem very sensitive to small changes in the alternative variables. Thus, it was suggested that the probability for every gene to be mutated was added, so that it would be possible to decrease the mutation by decreasing the value of the setting p_m. The variable $r = \left(r_1 \; \ldots \; r_{n_p} \right)$ is a random vector, its coordinates are randomly distributed for every offspring, $P\left(r_j = 0\right) = 1 - p_m, P\left(r_j = 1\right) = p_m$. Now the mutation operator can be described as follows, $j = \overline{1, n_p}$, $j_m = \overline{n_p + 1, 2 \cdot n_p}$

$$\left(i_m\right)_j = \left(i_c\right)_j + r_j \cdot N\left(0, \left(i_c\right)_{j+n_p}\right), \qquad (12)$$

$$\left(i_m\right)_{j_m} = \left(1 - r_j\right) \cdot \left(i_c\right)_{j_m} + r_j \cdot \left(\left(i_c\right)_{j_m} \cdot e^{\tau \cdot N(0,1)} \right), \qquad (13)$$

or

$$\left(i_m\right)_{j_m} = \left| \left(i_c\right)_{j_m} + r_j \cdot \tau \cdot N(0,1) \right|, \qquad (14)$$

where $\left(i_m\right)_j, j = \overline{1, n_p}$ are objective parameters, $\left(i_m\right)_{j+n_p}, j = \overline{1, N}$ are strategic parameters, τ is a learning coefficient, $N\left(E, \sigma^2\right)$ is a normally distributed random value with an expected value E and a variance σ^2.

The third modification is also supressing the influence of the random search on the order estimation. The vector determines the LDE order; therefore, some of its coordinates are equal to zero. However, we still need an operator to increase its accuracy, but we do not want this operator to influence on the order estimation. This leads to a contradiction. To solve it, a rounding operator was implemented. One more algorithm parameter sets the threshold level $0 \le t_l \le 1$, so the rounding operator works as follows

$$\left(i_m\right)_j = \begin{cases} \left(i_m\right)_j, if \left| \left(i_m\right)_j \right| > t_l \\ 0, otherwise \end{cases}, \; j = \overline{1, N_c}, \qquad (15)$$

where $N_c = M + 1$ the number of objective parameters that transform into ODE coefficients.

All the designed implementations sufficiently increased the efficiency of the algorithm and improved its performance. By implementing the modifications one by one, it can be seen that the efficiency of the algorithm is increasing with every implementation with the same computational resources. All the modifications were designed to lessen the complexity coming from the alternative-to-model transformation and the simultaneous estimation of LDE parameters.

3 RESTART OPERATOR

The next thing required is to avoid the stagnation of the algorithm, which is why a criterion to estimate whether stagnation has happened is needed. Firstly, understanding of algorithm behaviour requires some observations of the algorithm characteristics.

In this study we propose monitoring the fitness function value of the best solution found by the algorithms. This makes it possible to control when the optimization process is stopped if there is no improvement of accuracy. If the algorithm does not improve the best solution ever found and this value does not change for some number of iterations, one can conclude that there is a stagnation of the populations. Average fitness statistics do not allow an estimation to be made of whether the solution is changing or not.

Let $L(j,n_t)=\left\{f_{best}^i, i=\overline{j,j-n_t}\right\}$ be the set of fitness function values f_{best}^j of the best solutions ever found by the algorithm until the current generation j, n_t is the number of generations this set holds. It is easy to see that $\forall j_1, j_2 < n_t : j_1 > j_2 \Rightarrow f_{best}^{j_1} \geq f_{best}^{j_2}$ and in another notation $L_{j_1} \geq L_{j_2}$. Now we put forward a criterion for restarting based on these statistics:

$$C_1^r(j)=\max\left(L(j,n_t)\right)-\min\left(L(j,n_t)\right), \quad C_1^r(j)<c_1^r, \tag{16}$$

where n_t is the set size and $1 > c_1^r > 0$ is a threshold and both are controlling parameters.

Now let $x_{best}^k \in R^{n_p}$ be the best solution for which the algorithm was restarted according to criterion (16) and k is the number of restarts made. The set $H=\left\{x_{best}^i, i=\overline{1,k}\right\}$ consists of all these solutions. So if during the iteration process the algorithm restarts according to condition (16), then $k=k+1$, $H=H\bigcup x_{best}$. On the basis of the set H we put forward another criterion for restarting:

$$C_2^r=\min_{1\leq i\leq k}\left(\left\|x_{best}-H_i\right\|\right), C_2^r<c_2^r. \tag{17}$$

So with these heuristics we prevent both the stagnation of the populations – (16), and convergence to a stagnation area – (17).

To provide the maximum gain in efficiency with the implementation of the meta-heuristic, it is necessary to estimate the influence of settings on the algorithm characteristics.

Here we describe all the characteristics that are useful for the described identification problem. The first characteristic is the criterion (8), which we will denote as C_1, the value of the solution found by the algorithm. However, as the previous work shows, this is not the only reliable characteristic, since there is other knowledge that can be mined comparing the solution with the real dynamical system. The following metric can be evaluated:

$$C_2=\frac{h_s}{2}\cdot\sum_{i=1}^{N_s-1}\left|x(t_i)-\hat{x}(t_i)\right|+\left|x(t_{i+1})-\hat{x}(t_{i+1})\right|, \tag{18}$$

where h_s is the step size and $N_s=\frac{t_s}{h_s}$ is the number of steps. The value of this criterion shows the closeness of the model output to the real output trajectory. It is useful in estimating the quality of the solution for a noised sample.

The next characteristic that interests us is the probability of identifying the real order. We can estimate this probability with the following criterion:

$$C_3=N_{success}/N_{runs}, \tag{19}$$

where $N_{success}$ is the quantity of solutions with the correct order of the differential equation and $N_{launches}$ is the quantity of algorithm runs.

If the order is identified correctly, the distance between the model parameters and the real system parameters can be calculated as

$$C_4=\left\|a-\hat{a}_m\right\|+\left\|x_0-\hat{x}_0^m\right\|, \tag{20}$$

here we suggest using the Euclidian norm.

To make an adequate estimation of the algorithm, more than only criteria (8), (18) and (20) is required. We suggested adding two more criteria, having considered of all the options. If the sample size is small and measurements are distorted, the criteria described above indicate the complexity of the problem only, because the solution could fit the observations more than a real system output trajectory.

Let $C_s=I(a,x_0)$ be a criterion (8) value for real dynamical system parameters. We put forward criterion

$$C_5 = N\left(C_1(\hat{a},\hat{x}_0) < C_s\right)\cdot\left(N_{runs}\right)^{-1}, \tag{21}$$

$N\left(C_1(\hat{a},\hat{x}_0) < C_s\right)$ is the number of event $C_1(\hat{a},\hat{x}_0) < C_s$ that has occurred. Furthermore, if the event occurs, the value that would show how much better the model is can be calculated:

$$C_6 = C_s - C_1(\hat{a},\hat{x}_0)\,. \tag{22}$$

By varying the meta-heuristic settings and running the algorithm 25 times for every distinct identification problem, we can estimate the most efficient settings.

4 PERFORMANCE INVESTIGATION

Let us briefly discuss the sample forming process. The dynamic system output is the Cauchy problem solution on $[0,T]$, as it corresponded to (1). Then the solution is discretised and represented as a set with N_s elements. Let $I^s = \left\{r_i^{\text{int}} \le N_s, \forall i < s\right\}$ be a set of randomly chosen different integers, so accordingly, the expression (2) is, $y_i = x\left(I_i^s \cdot T/N_s\right) + N\left(0,\sigma^2\right)$ and $t_i = I_i^s \cdot T/N_s$, where $i = \overline{1,s}$.

To examine the algorithm with meta-heuristics and its different settings, three different identification problems were considered. The list of differential equations that simulated the dynamic process is given in Table 1. The sample size is 100. For this experiment the variance of the noise is equal to 0, so the experimental data allows us to see the algorithm performance in a basic form.

Table 1: Identification problems.

Identification problems	
1	$x'' + 2\cdot x' + x = u(t)$, $x_0 = (2\ 0)^T$, $T = 12.5$
2	$x''' + x'' + 2\cdot x' + x = u(t)$, $x_0 = (2\ 0\ 0)^T$, $T = 12.5$
3	$x^{(4)} + x''' + 4\cdot x'' + 3\cdot x' + x = u(t)$, $x_0 = (2\ 0\ 0\ 0)^T$, $T = 12.5$

Due to the results of previous works it was decided to use the following settings in the current investigation: 100 individuals for 100 populations, tournament selection, 3 parents for random crossover, mutation scheme (12) and (14), the mutation probability $p_m = 2/n_p$, $N_1^L = 40$ individuals for the local stochastic improvement, $N_2^L = 50$ genes and $N_3^L = 1, h_L^{\max} = 0.5$, the threshold for rounding $t_l = 0.4$. The strategic parameters of the initial population were uniformly generated $U(0,1)$, and every objective parameter is equal to 0. The order limitation value took 10.

The parameters of the restart operator took values from the following sets: $n_t \in \{5, 7, 9, 11, 15\}$, $c_1^r \in \{0.005, 0.01, 0.05, 0.1\}$, $c_2^r \in \{0.05, 0.1, 0.2, 0.5\}$, and in this investigation we checked all the possible combinations of parameter values. We compared algorithms using the average values of criteria (8), (18)-(20). The algorithm with minimum error (8) and (18) is the one with the settings $c_1^r = 0.005$, $c_2^r = 0.05$, $n_t = 9$, and the criterion average value equal to 0.0037. The algorithm with the maximum average number of instances of estimating the probability of identifying the correct order is the one with the settings $c_1^r = 0.05$, $c_2^r = 0.1$, $n_t = 15$, the average criterion (19) value is 0.9066. And this algorithm complies with the minimum average value of criterion (20), which is equal to 0.0609.

Let us show how the examination results correlate with the performance estimation of the hybrid modified evolutionary strategies algorithm without the meta-heuristic. For this purpose we put on a chart the average criterion (8) value for the algorithm without the meta-heuristic, the average criterion value for the algorithm with the best meta-heuristic settings and the same for the worst, which is marked as HMES, HMES+best and HMES+worst, respectively. We put the same statistics on the chart for criteria (18)-(20). The first two charts are given in Figure 1 and the second in Figure 2.

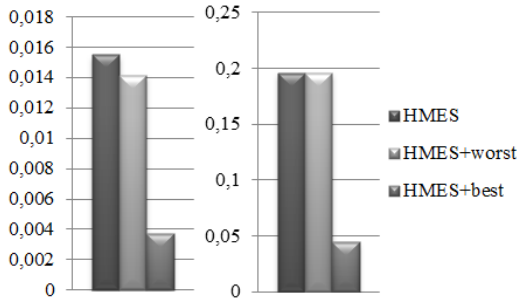

Figure 1: The average error of the observation data fitting(left) and the average error of the model output with the real system trajectory(right).

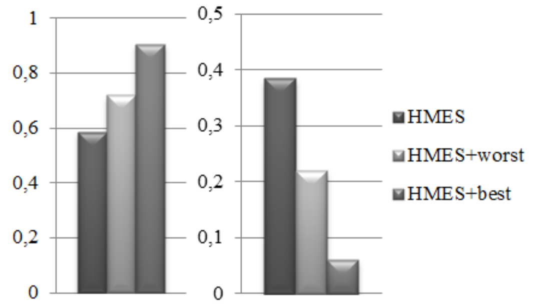

Figure 2: The average probability of receiving a model with the right order values(left) and the average error of the real parameter estimation(right).

Now we estimate and compare the algorithm performances for solving the identification problems with different sample sizes and noise variances. The restart operator settings were taken as follows: $c_1^r = 0.005$, $c_2^r = 0.05$, $n_t = 9$.

For identification problems 1-3, given in Table 1, the sample size takes: 200, 150, 80 and 40; the noise parameter n_l takes: 0.05, 0.1, 0.2, 0.5 and 0.8, we examine the algorithm by running it on each of the considered combination.

Comparing the performances of HMES and HMES+best gives us the following results: the average criterion (8) values are 0.0692 and 0.0632, respectively; the average criterion (18) values are 0.4719 and 0.3732, respectively. The proposed algorithm finds a model that fits the observations more than the real object trajectory with average probability 0.8133, (21). This shows that the algorithm with the restart operator is more efficient on average. Moreover, in every distinct experiment its average fitness value was greater than the average fitness value of HMES. The average criterion (18) values are shown in Figure 3. The chart is divided into 4 parts with vertical lines and each part relates to one of the sample size: from the left – 200, to the right – 40. In every part the bars start with the problems with $n_l = 0.05$ and go to the right, up to $n_l = 0.8$.

As can be seen, in some experiments, the proposed algorithm provided a worse result, but it found a better solution according to the main criterion (8). The criterion (18) values of the proposed algorithm are statistically linear to the noise level, so we can conclude that it is really more powerful in estimating a model that fits the observed data.

The same estimation of algorithm performance was made for the algorithm with the restart operator settings: $c_1^r = 0.05$, $c_2^r = 0.1$, $n_t = 15$. The average

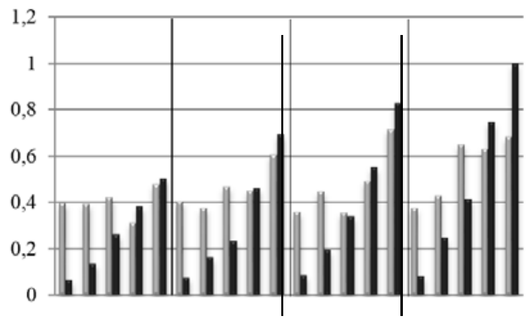

Figure 3: The average criterion (19) values for different sample sizes and noise variances: HMES (light grey) and the algorithm with meta-heuristic (black).

values of its criteria are better than the same values of the HMES algorithm, but worse than the values of the previous algorithm: the average criterion (8) value is 0.0643, the criterion (18) value is 0.3822. This algorithm finds a better solution than the real system trajectory with a probability of 0.8433.

On the basis of the examination performed we can conclude that the settings do not save the properties of the algorithm if it is applied to a problem with distorted data. For example, the probability of identifying the correct differential equation order with the second algorithm is 0.6313, and with the first – 0.6446; for different sample size and without the noise the second algorithm was the best according to this criterion.

5 CONCLUSIONS

The examination of the approach proposed in this paper shows that the designed meta-heuristic tends to improve algorithm performance. This occurs even though the basic algorithm is already problem-oriented: it is hybridized, modified and more efficient than the standard evolution-based algorithms applied to this problem. The important fact is that the improvement in performance occurs in any case, whatever values from the given sets the parameters take.

The aim of the meta-heuristic is to improve the algorithm efficiency in reaching a global optimum neighbourhood by monitoring the algorithm characteristics and collecting information about the previous behaviour of the algorithm. There are obviously many different possibilities of how to achieve this and this is the purpose of further studies.

In this study we examined the algorithm with the most efficient restart operator parameters according to various criteria. For each criterion we received a preferable size of the memory set and radiuses of the neighborhoods for detecting the stagnation and for detecting if the solution is going towards one that has been already found.

Further work is related to examining the influence of the restart settings on the algorithm performance in the case of different sample sizes and noise variances in order to discover more robust settings.

ACKNOWLEDGEMENTS

Research is performed with the financial support of the Russian Foundation of Basic Research, the Russian Federation, contract №16-01-00767, dated 03.02.2016.

REFERENCES

Beligiannis G. N., Tsirogiannis G. A., Pintelas P. E., 2004: Restartings: A Technique to Improve Classic Genetic Algorithms' Performance. *J. Glob. Optim*. 1, pp. 112-115.

Dao S. D., Abhary K., Marian R. M., 2014: Optimization of partner selection and collaborative transportation scheduling in Virtual Enterprises using GA. *Expert Syst. Appl*. 41(15), pp. 6701-6717.

Eiben A. E., Smith J., 2015: From evolutionary computation to the evolution of things. *Nature*, 521(7553), pp. 476–482.

Fukunaga A. S., 1998, Restart Scheduling for Genetic Algorithm. *Lecture Notes of Computer Science*, *Vol*. 1498, pp. 357-369.

Loshchilov I., Schoenauer M., Sebag M., 2012: Alternative Restart Strategies for CMA-ES, *Lecture Notes in Computer Science*, *Vol*. 7491, pp. 296-305.

Naiborhu J., Firman, Mu'tamar K., 2013. Particle Swarm Optimization in the Exact Linearization Technic for Output Tracking of Non-Minimum Phase Nonlinear Systems. *Applied Mathematical Sciences*, *Vol. 7*, *no. 109*, pp. 5427-5442.

Narwal A., Prasad B. R., 2016. A Novel Order Reduction Approach for LTI Systems Using Cuckoo Search Optimization and Stability Equation. *IETE Journal of Research*, 62:2, pp. 154-163.

Parmar G., Prasad R., Mukherjee S., 2007. Order reduction of linear dynamic systems using stability equation method and GA. *International Journal of computer and Infornation Engeneering* 1:1, pp.26-32.

Ryzhikov I., Semenkin E., Panfilov I., 2016. Evolutionary optimization algorithms for differential equation parameters, initial value and order identification. *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics* (*ICINCO* 2016) - *Volume* 1, pp. 168-176.

Ryzhikov I., Semenkin E., 2013. Evolutionary Strategies Algorithm Based Approaches for the Linear Dynamic System Identification. *Adaptive and Natural Computing Algorithms. Lecture Notes in Computer Science*, *Volume* 7824. – Springer-Verlag, Berlin, Heidelberg, pp. 477-483.

Schwefel Hans-Paul, 1995. *Evolution and Optimum Seeking*. New York: Wiley & Sons.

Sersic K., Urbiha I., 1999. Parameter Identification Problem Solving Using Genetic Algorithm. *Proceedings of the 1. Conference on Applied Mathematics and Computation*, pp. 253–261.

Wolpert D.H., Macready, W.G., 1997: No Free Lunch Theorems for Optimization, *IEEE Transactions on Evolutionary Computation*, 1, pp. 67-82.