Quantitative Finance and Economics

http://www.aimspress.com/journal/QFE

QFE, 3(4): 608–623
DOI:10.3934/QFE.2019.4.608
Received: 27 July 2019
Accepted: 19 September 2019
Published: 30 September 2019

Research article

BOMD: Building Optimization Models from Data (Neural Networks based Approach)

Vladimir Donskoy*

Department of Mathematics & Computer Science, V.I. Vernadsky Crimean Federal University, Simferopol, 295007, Russia

* **Correspondence:** Email: vidonskoy@mail.ru; Tel: +79787340290; Fax: +73652608498.

Abstract: This article aims to develop mathematical methods and algorithms that automatically build nonlinear models of planning and management of economic objects based on the use of empirical samples (observations). We call the relevant new information technology "Building Optimization Models from Data (BOMD)". The offered technology BOMD allows to obtain an objective control models that reflect the real economic processes. This is its main advantage over commonly employed subjective approach to management. To solve the problems posed in the article, the methods of artificial intelligence were used, in particular, the training of neural networks and construction of decision trees. If the learning sample contains simultaneously the values of the objective function and the values of characteristic function of constraints, it is proposed to use an approach based on the training of two neural networks: NN1 — for the synthesis of the objective function and NN2 — for the synthesis of the approximating characteristic function of constraints (instead of a neural network NN2, a decision tree can be used). The solution of the problem presented by such synthesized neural model may end up finding, generally speaking, a local conditional extremum. To find the global extremum of the multiextremal neural objective function, a heuristic algorithm based on a preliminary classification of the search area by using the decision tree is developed. Presented in the paper approach to an extraction of conditionally optimization model from the data for the case when there is no information on the points not belonging to the set of admissible solutions is fundamentally novel. In this case, a heuristic algorithm for approximating the region of admissible solutions based on the allocation of regular (non-random) empty segments of the search area is developed.

Keywords: intelligent economic management; model extraction; sample; neural network; decision tree
JEL Codes: C45, D22, D83, M11

1. Introduction

In this paper, it is shown how can be synthesized models of optimal control or planning based on samples or precedents. The proposed BOMD (Building Optimization Models from Data) approach is based on empirical induction and directed to obtaining regularities in the form of empirical optimization models which are synthesized in analytical form. We follow the Kolmogorov idea about regularity as non-randomness. This allows us to estimate the probability of non-random model selection from the set of admissible models that are consistent with the sample or to given initial data. The proposed methods and algorithms can be applied to solve a wide range of tasks of intelligent control and planning, in particular, production planning and robotics. Although the control models derived from the data are approximate, their application can be more successful than the use of less realistic, inconsistent with the objects models which are chosen on the base of subjective considerations.

Unfortunately, very few scientific papers are devoted to the theory of *optimization models extraction from data*. The main results in this direction are presented mainly in Mazurov (1971); Eremin and Mazurov (1979); Donskoy (1993, 2017, 2018). In broad terms, problems considered in this paper are included in the scope of problems acquisition of optimization models from data and decision making with incomplete initial information. Such problems were the first investigated by Vl.D. Mazurov Mazurov (1971) on the base of theory of pattern recognition which is an essential element of the BOMD paradigm Donskoy (1993); Antonova (2011). In recent years, interest in the subject has increased due to the development of cognitive technologies and big data technology, especially in interests to business Barton D and Court D (2013); MathWorks (2017). Previously, the author obtained results in the BOMD domain for the case of linear Donskoy (1993) and pseudo-Boolean models Donskoy (2018).

This paper continues the research within the paradigm of extracting or building optimization models from data for intelligent control systems. The obtained results are devoted to nonlinear models with real variables, generally speaking, of any functional complexity in the class of functions of arbitrary degree of smoothness and constraints represented by piecewise linear approximation. This is achieved through the use of neural networks as the main used mathematical apparatus. If the learning sample contains simultaneously the values of the objective function and the values of characteristic function of constraints, it is proposed to use an approach based on the training of two neural networks: $NN1$ — for the synthesis of the objective function and $NN2$ — for the synthesis of the approximating characteristic function of constraints. Unfortunately, the solution of the problem defined by such 2-neural synthesized model may end up finding, generally speaking, only a local conditional extremum. To find the global extremum of the multiextremal neural objective function, a heuristic algorithm based on a preliminary classification of the search area by using the decision tree is developed. The presented in the paper approach to an extraction of conditionally optimization model from the data for the case when there is no information on the points not belonging to the set of admissible solutions is fundamentally novel. In this case, a heuristic algorithm for approximating the region of admissible solutions based on the allocation of regular (non-random) empty segments of the search area is developed. When using this approach in practice in intelligent control systems, it is necessary to additionally apply human-machine procedures for verification and correction of synthesized models.

Let $X^n = X_1 \times \cdots \times X_i \cdots \times X_n$ is limited area in R^n which is called the space of variables-features

of dimension n; $\tilde{x} = (x_1, \cdots, x_i, \cdots, x_n)$ — an arbitrary point in the space of variables which is a description of an admissible object.

Each coordinate x_i, $i = 1, \ldots, n$, of the object description $\tilde{x}$ belongs to some fixed *limited set of admissible values* X_i, $X_i \subset \mathrm{R}$, $m_i \le x_i \le M_i$.

$\mathrm{T}_{Opt} = \{(\tilde{a}_j, \gamma_j, y_j)\}_{j=1}^{l}$ is a reliable empirical sampling for the *especial machine learning problem – model extraction based on partial information about some existing but unknown scalar criterion* $F : X^n \to \mathrm{R}$ *and unknown constraints that can formally be represented as* $\Omega(\tilde{x}) = 1$.

In the problem under consideration denoted below $Z_{\Omega,F}$, it is believed that the set X^n is divided into two classes: the class L_1 consisting of points which satisfy a system of the constraints in the problem of the best choice (termed admissible points), and the class L_0 contains the points which is knowingly not satisfy the system constraints. We denote $\Omega : X^n \to \{0; 1\}$ – *the characteristic function of the constraints, which is partially defined by the learning sample*; $\Omega(\tilde{x}) = 1 \Leftrightarrow \tilde{x} \in L_1$; $\Omega(\tilde{x}) = 0 \Leftrightarrow \tilde{x} \in L_0$. We assume that standard learning sample T_{Opt} contains reliable information $\gamma_j = \Omega(\tilde{a}_j)$, $\gamma_j \in \{0; 1\}$, $y_j = F(\tilde{a}_j)$, $\tilde{a}_j = (a_{j1}, \ldots, a_{jn}) \in \mathrm{R}^n$.

In the learning process, we should build a rule (algorithm) that allows to choose such a solution $\hat{\tilde{x}}^*$ which would satisfy the constraints ($\Omega(\hat{\tilde{x}}^*) = 1$) and the value $F(\hat{\tilde{x}}^*)$ would be as big as possible (or less — in the sense of the assigned problem).

In the problems under consideration the criterion F and the constraints (*characteristic function* Ω) *are not given exactly — neither analytically, nor completely tabular, nor using of any formal system. They are "reflected" by the values of the learning sample* T_{Opt} *and are only partially defined.*

Statement of the problem solved in this paper is as follows. It is required (using partial initial information T_{Opt}) choose a solution $\hat{\tilde{x}}^*$ that is as close as possible to optimal solution $\tilde{x}^*$ determined unknown but true exists objects F and Ω, which are approximated by neural networks*. If the scalar criterion and the characteristic function of the constraints are approximated independently by separate algorithms that compute as accurate as possible in any sense of the approximation $\hat{F}$ and $\hat{\Omega}$ then *restored from the learning sample problem* (*extracted mathematical model*) *finding the best solution is as follows*:

$$max(min)\ \hat{F}(\tilde{x}) : \hat{\Omega}(\tilde{x}) = 1 \wedge \tilde{x} \in X^n.$$

A pair of functions $< \hat{F}, \hat{\Omega} >$ resulting from machine learning is called *empirical information model*.

The idea of BOMD paradigm to facilitate its understanding can be illustrated by the following simple scheme (Figure 1). In the process of observation of the object and the environment, numerical data on its current characteristics - parameters (features), as well as the values characterizing the quality of operation of the object, must be recorded in the computer's memory. Besides, the facts of being in unacceptable (wrong) state the object is recorded. Such wrong object states are defined by experts-observers. The data obtained are grouped into training samples that reflect the aggregate of valid and invalid object states and quality values for each fixed set of parameters. Then, by performing empirical generalization using machine learning, the synthesis of the empirical information model is carried out.

*In some cases, classifying trees will be used to approximate the Ω domain.

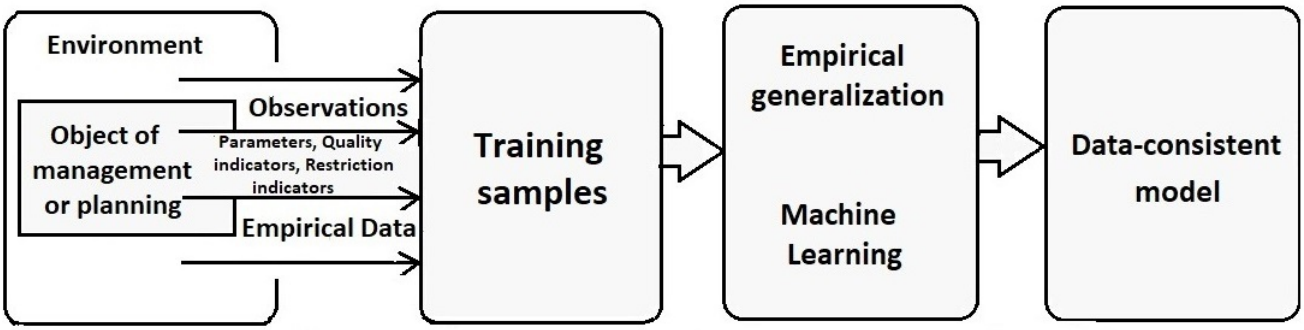

Figure 1. BOMD paradigm idea.

We emphasize the main advantage of the BOMD paradigm: the optimization empirical model is derived from real data about the control object, and it is not subjective proposed by some expert. In contrast, a subjective model that is not consistent with real information can lead to decisions that do not correspond to reality, although mathematically can be calculated accurately.

Applications BOMD technology is very various. It is known how widely used mathematical models of optimization in economics and engineering. This is because each production and technical system aims to maximize quality or profit (or minimize costs).

Having an empirical optimization model, it is possible to implement a controller of any technical device. The control process is realized as shown in the Figure 2. The main element of the controller is this optimization model which is synthesized from data. The coordinates of the vector $\hat{\hat{x}}^*$ are the control actions $x_1, ..., x_n$.

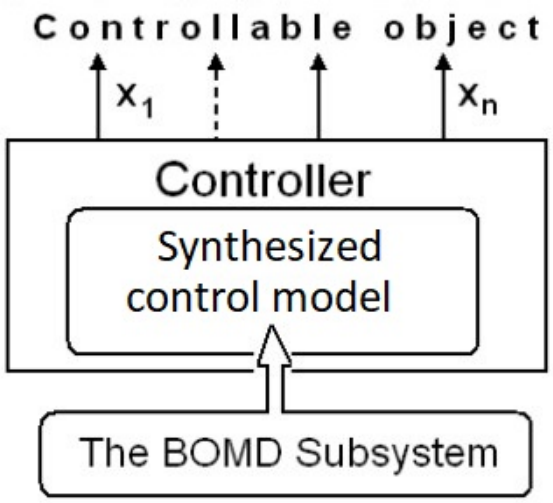

Figure 2. BOMD-Model based controller.

It should be noted, that for some production systems it is extremely difficult and even impossible to subjectively write out a mathematical model of management. For example, for multi-machine production with a large number of products and processes. BOMD technology also may be successfully applied in this case.

One example of the practical implementation of the proposed approach is presented below (see Section 5, Numerical experiment).

The use of BOMD technology involves a lot of preparatory work for the collection of initial information and the formation of training samples. Despite the importance of these preparatory actions, their description is not the purpose of this article and the main problem on which the focus pays attention. Principles of partition of initial data (percentage of training, validation, etc.) remain outside the scope of this paper. These issues are widely covered in publications, for example, Roh et al. (2019); Zhang et al. (2003).

2. The principles of extraction of neural optimization models from data. The case where the constraints are partially set by precedents in the learning sample

We assume that by using precedents describing unknown objective function F, we'll synthesize an approximating neural network[†] $NN1$ for implementation the function $\hat{F}_{NN1}(\tilde{x}) = \mathcal{F}(\tilde{x})$, i.e. $\mathcal{F}$ is an approximation for F. And by precedents describing the constrains the classifying neural network denoted $NN2$ will be learned to approximate the characteristic function of constraints $\Omega(\tilde{x})$: $\hat{\Omega}_{NN2}(\tilde{x})$. For convenience following calculations, we present a multilayer fully connected approximating neural network $NN1$ by entering an additional "hidden" layer with the number zero (Fig.3).

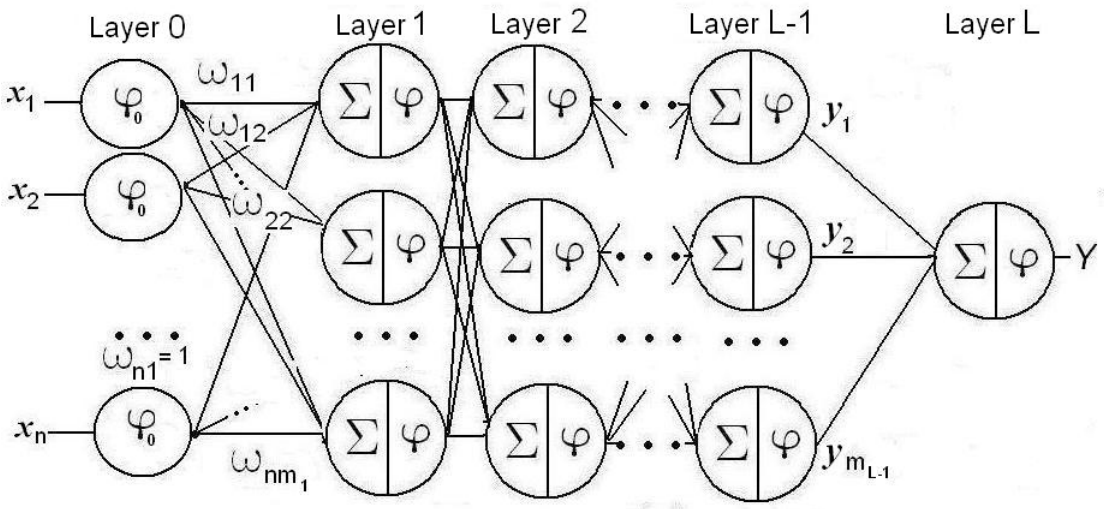

Figure 3. A network $NN1$ with conditionally added layer number zero. In order not to complicate the drawing, offsets (biases) — an extra inputs to neurons — are not shown.

"Activation functions" of the layer 0 of the form $\varphi_o(z) = z$ are added for the convenience computations.

The activation functions in all other layers will be considered logistic:

$$\varphi(z) = \frac{1}{1+e^{-z}};\ \varphi'(z) = \varphi(z)(1-\varphi(z)). \tag{1}$$

[†]Neural networks are now so widely used in various fields of economics and technology that we can even talk about a excessive "boom" in this direction. Therefore, it seems unnecessary to include a section on neural networks in this article. For those who are not familiar with the above subject, one can recommend the article Abiodun et al. (2018).

We will use the following notation:

v_j – weighted sum of all neuron j inputs which is called its induced local field Haykin (2008);

l – layer number, $0 \le l \le L$; a layer number 0 is a special input layer; output layer has number$L$;

$m_1, m_2, \ldots, m_{L-1}$ — number of neurons in hidden layers $1, 2, \ldots, L-1$;

$\tilde{x} = (x_1, \ldots, x_i, \ldots, x_n)$ — the input of neural network;

$y_j = \varphi(v_j)$ — the output of neuron j; $Y = Y(\tilde{x})$ — the network output.

To find the gradient of the function implemented by the trained neural network presented in Figure 3 we will use a recursive scheme which underpins the learning algorithm of the neural network by the method of backpropagation of error Haykin (2008).

The local gradient of the output neuron is determined by the expression

$$\delta_L^{(L)} = \frac{\partial Y}{\partial v_L} = \varphi'(v_L) = \varphi(v_L)(1 - \varphi(v_L)) = Y(1 - Y), \tag{2}$$

where the superscript in parentheses indicates the network layer number.

Local gradient of neuron j of hidden layer with number l:

$$\delta_j^{(l)} = \varphi'(v_j) \sum_k \delta_k^{(l+1)} \omega_{jk} = y_j^{(l)}(1 - y_j^{(l)}) \sum_k \delta_k^{(l+1)} \omega_{jk}, \tag{3}$$

where the sum is taken from all the neuron numbers of the layer immediately following the layer containing the neuron j.

$$\delta_j^{(1)} = \varphi'\Big(\sum_{i=1}^{n} x_i \omega_{ij}\Big) \sum_k \delta_k^{(2)} \omega_{jk} = y_j^{(1)}(1 - y_j^{(1)}) \sum_k \delta_k^{(2)} \omega_{jk}. \tag{4}$$

$$\delta_i^{(0)} = \varphi_0'(x_i) \sum_k \delta_k^{(1)} \omega_{ik} = \sum_k \delta_k^{(1)} \omega_{ik}. \tag{5}$$

$$\frac{\partial Y}{\partial x_i} = \delta_i^{(0)}; \;\; grad\, Y = (\delta_1^{(0)}, \ldots, \delta_n^{(0)}). \tag{6}$$

As with the backpropagation calculations, for a given input $\tilde{x}$, the local fields and outputs of all neurons are first computed in the forward direction from the input to the output of the network. Then, in the opposite direction, starting with the equation (2), the calculations of local gradients by the formula (3) are performed recursively and completed by the calculation of the gradient by the formulas (5) and (6).

Finding the extremum of the empirical information model $< \hat{F}_{NN1}, \hat{\Omega}_{NN2} >$ can be done by gradient algorithm 1 presented below. In general, given that the $\hat{F}_{NN1}$ approximation may be multiextremal, this algorithm will look for a local extremum.

Algorithm 1. Search for conditional (local) minimum by $\hat{F}_{NN1}$ and $\hat{\Omega}_{NN2}(\tilde{x})$.
Input: Learning sample $\mathrm{T}_{Opt} = \{(\tilde{a}_j, \gamma_j, y_j)\}_{j=1}^{l}$, neural approximations $\hat{F}_{NN1}$, and $\hat{\Omega}_{NN2}(\tilde{x})$.
Output: $\tilde{x}^* : \hat{F}_{NN1}(\tilde{x}^*) \approx \min \hat{F}_{NN1}(\tilde{x})$ / $\hat{\Omega}_{NN2}(\tilde{x}) = 1$ – the point of extremum of empirical information model and value y^* of the function $\hat{F}_{NN1}$ at this point.

1: Take from the training sample a point $\tilde{x}_0 = \tilde{a}_{j^*} : \; y_{j^*} = \min_j y_j$ as an initial approximation and calculate $\hat{F}_{NN1}(\tilde{x}_0)$.
2: Choose the initial value η_0 and the accuracy of the approximation ε.
3: Choose a learning rate reducing coefficient $\delta : \; 0,8 < \delta < 1$.
4: **for** $t := 1, 2, 3, \ldots$ **do**
5: $\quad \tilde{x}_t := \tilde{x}_{t-1} - \eta_{t-1} \, grad \, \hat{F}_{NN1}(\tilde{x}_{t-1})$;
6: $\quad \eta_t := \eta_{t-1} \cdot \delta$;
7: $\quad$ **if** $\hat{\Omega}_{NN2}(\tilde{x}_t) = 0$ **then**
8: $\quad\quad$ **{ if** $\rho(\tilde{x}_t, \tilde{x}_{t-1}) < \varepsilon$ **then goto** 12 **else** $\eta_t := \eta_{t-1} \cdot \delta$ **}**
9: $\quad$ **else**
10: $\quad\quad$ **if** $\|\hat{F}_{NN1}(\tilde{x}_t) - \hat{F}_{NN1}(\tilde{x}_{t-1})\| < \varepsilon$ **then goto** 13;
11: **end for**;
12: $\tilde{x}^* := \tilde{x}_{t-1}; \; y^* := \hat{F}_{NN1}(\tilde{x}_{t-1})$; **stop.**
13: $\tilde{x}^* := \tilde{x}_t; \; y^* := \hat{F}_{NN1}(\tilde{x}_t)$; **stop.**

3. Heuristic approaches to finding the global extremum of the model $< \hat{F}_{NN1}, \hat{\Omega}_{NN2} >$

A search for the global extremum for the problem under consideration is based on the repeated application of the above algorithm 1 for finding a local extremum starting from different initial points of the domain of admissible solutions.

The First, traditional approach based on the application of genetic algorithms, the description of which can be found in extensive scientific and technical literature El-Sawi et al. (2014), will not be adapted to the problem. The genetic algorithm is a heuristic search algorithm used to solve optimization problems by random selection, combination, and variation of the desired parameters using mechanisms similar to natural selection in nature Malhotra et al. (2011). Genetic algorithms are members of the class of evolutionary algorithms Corne and Lones (2018), which are based on the principles of population genetics. To solve the problem of approximate search of the global extremum of the model $< \hat{F}_{NN1}, \hat{\Omega}_{NN2} >$, it can be used a different approach, described below, which will allow to better use the regularities reflected in the training sample, and not to use random selection, combinations and variation of local solutions.

This second, new approach outlined below, is based on the preliminary clustering of the points from the sample $\mathrm{T}_{Opt} = \{(\tilde{a}_j, \gamma_j, y_j)\}_{j=1}^{l}$ by the values of the objective function $y_j, \; j = \overline{1, l}$. For this purpose, an algorithm of building a decision tree, defining the partitioning of the region of the admissible values of the variables into hyperparallelepipeds is used Breiman L, Friedman JH, Olshen R, et al. (1984). In the region corresponding to each such hyperparallelepiped, the values of the objective function y_j belong to the fixed semi-interval. The number of such semi-intervals is the number of classes in the preliminary clustering problem. After constructing the classifying tree each class receives a logical description in terms of threshold predicates of the form $[x_i \le b]$ where b is a real constant, $i \in \{1, ..., n\}$.

It is necessary to determine two constants m and $\mathcal{M}$ such that $m < y_j < \mathcal{M}$ for all $j = \overline{1,l}$ from the content considerations determined by the problem domain of the problem. Then divide the segment $[m, \mathcal{M}]$ into k equal segments $[m, m+\Delta), [m+\Delta, m+2\Delta) \ldots, [m+k\Delta, \mathcal{M}]$, where $\Delta = (\mathcal{M} - m)/k$. If the point $\tilde{a}_j$ of the learning sample falls into the segment with the number q, $q = \overline{1,k}$, this point is considered to belong to the class $\mathcal{K}_q$. Thus, in each of the obtained classes, there will be points where the values of the objective function differ by no more than Δ.

Denote $\bar{y}_q$ – the average value (a mid of q-th interval) $\bar{y}_q = m + q\Delta - \Delta/2$. Each end vertex of the tree (leaf) contains the value $\bar{y}_q$ corresponding to the class number q and a special label-flag used to remember the viewed leaves when searching for the global extremum.

Algorithm 2. Heuristic search for a conditional global minimum of an empirical information model $< \hat{F}_{NN1}, \hat{\Omega}_{NN2}(\tilde{x}) >$.

Input: a learning sample $\mathrm{T}_{Opt} = \{(\tilde{a}_j, \gamma_j, y_j)\}_{j=1}^{l}$; neural approximations $\hat{F}_{NN1}$ and $\hat{\Omega}_{NN2}(\tilde{x})$; the algorithm 1 for finding a local minimum as used internal procedure; the point clustering tree by target values of objective function as the second used internal procedure; the most allowed number of local search steps S.

Output: $\tilde{x}^* : \hat{F}_{NN1}(\tilde{x}^*) \approx \min \hat{F}_{NN1}(\tilde{x}) \,/\, \hat{\Omega}_{NN2}(\tilde{x}) = 1$ — the intended point of global extremum of empirical information model and value y^* of the function $\hat{F}_{NN1}$ at this point.

1: Clear labels in all the leaves of the tree (all labels will get null values).
2: Take a point $\tilde{x}_0 = \tilde{a}_{j^*} : \; y_{j^*} = \min_j y_j$ from the training sample as the initial value and execute the algorithm 1.
3: Remember the found point as $\tilde{x}^*$ and the value of the local minimum at it in variable y^*.
4: Mark a leaf of a tree in which the point $\tilde{x}^*$ was hit (set the label to one).
5: Select the unmarked leaf of the tree with the lowest value $\bar{y}_q$ and the point of the training sample contained in it with the smallest value of the objective function.
6: Apply algorithm 1 to the selected point to get the extremum $y^\diamond$ at the point $\tilde{x}^\diamond$.
7: Mark the selected leaf as viewed (check label to one).
8: **if** $y^\diamond < y^*$ **then** $\{y^* := y^\diamond; \tilde{x}^* := \tilde{x}^\diamond\}$;
9: If unlabeled leaves of the tree still exist and the number of leaf viewing (applications of the algorithm 1) did not exceed the given number S then goto 5;
10: The end of search.

The substantiation for the use of the classifying algorithm in the search for the global extremum in the algorithm 2 is as follows.

1. The number of l points in the training sample is many times the number of leaves in the classification tree. Therefore, if instead of iterating over all these l points to initialize and execute the local search, we use the above described classifying algorithm then the local search will be repeated no more than $\mu \ll l$ times – each time with a starting point taken from the segment corresponding to the selected tree sheet.

2. The leaves of the tree correspond to different, remote from each other segments[‡] of the global search area from which initial points for extremum finding are chosen, while other, for example, random way of choice of points from the whole training sample to initialize a local search may result in repeated calculations in a sufficiently narrow neighborhood of the same local minimum.

4. The case where there is no information about points that are not admissible (not belonging to the Ω area)

The collection of initial training information in solving problems in the management of large systems is no less difficult than the development of the necessary software, especially because in recent years, many systems and programming environments are equipped with powerful libraries that implement various methods of machine learning and decision-making.

It is particularly difficult to get data on points or states of optimizing or managed objects that are not admissible – do not satisfy system constraints. In our case, it is part of the standard learning information $\mathrm{T}_{Opt} = \{(\tilde{a}_j, \gamma_j, y_j)\}_{j=1}^{l}$ consisting of points $\tilde{a}_j$ such that $\gamma_j = \Omega(\tilde{a}_j) = 0$. If the data about the optimized object are accumulated in the process of its functioning, usually there are some admissible states of its operation; the other states are understood as "breakdown" ones and are not admissible. The occurrence of such a state can be considered a rare event.

The above considerations lead to the expediency of considering the case when the training information has the form $\mathrm{T}^{-}_{Opt} = \{(\tilde{a}_j, y_j)\}_{j=1}^{l}$ and it contains only admissible points for which $\gamma_j = \Omega(\tilde{a}_j) = 1$.

The idea of an approach to the construction of constraints for such a case is related to the allocation in the search area of the global extremum of the so-called *empty segments* in which did not hit the sampling points from T^{-}_{Opt} belonging to the area of admissible solutions. Figure 4 explains this approach. Search areas are conditionally shown in which asterisks denote admissible points and empty segments are denoted as $E_1, \ldots, E_6$.

Recall that in this paper we consider regular processes and objects. In this case, we are not talking about any probability distributions, but it is still possible to *estimate the selected empty segments based on the Kolmogorov approach to estimating the regularity as non-randomness*.

A. N. Kolmogorov emphasized the need to distinguish between actual *randomness as the absence of regularity* and a stochastic randomness as the subject of the theory of probability (Kolmogorov , 1991, p. 42). When empirical extraction of regularities the Kolmogorov approach allows us to estimate the non-randomness of the presence of an empty segment.

[‡]The distance between two regions W_1 and W_1 is understood as

$$\inf_{\{\tilde{x}\in W_1,\ \tilde{y}\in W_2\}} \rho(\tilde{x}, \tilde{y})$$

where ρ is Euclidean distance between points in R^n.

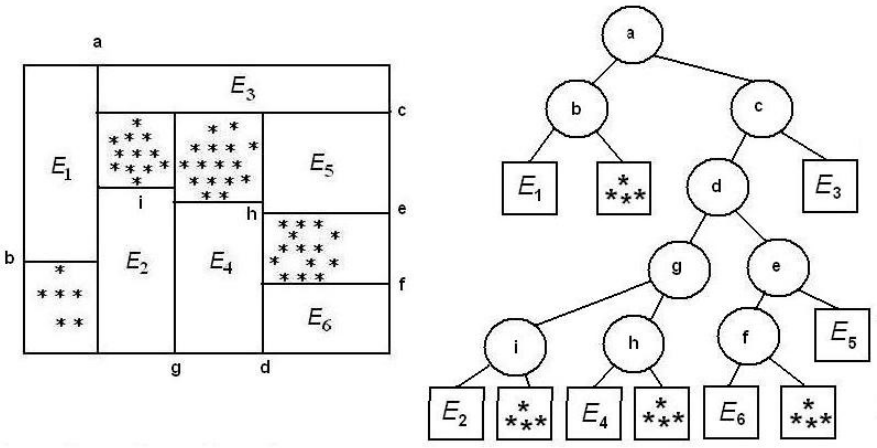

Figure 4. Empty segments and segments of the admissible solutions.

The absence of regularity in the arrangement of points in the area of possible values of variables in the form of hyperparallelepiped of the volume

$$\mathrm{V}_0 = \prod_{i=1}^{n} [m_i, M_i], \quad m_i \le x_i \le M_i, \tag{7}$$

takes a place when the points are distributed uniformly, randomly and independently. But if, for example, the projection of points a_j, $j = \overline{1,l}$, of the training sample to some coordinate x_i is represented by the histogram shown on the fig. 5 then the regularity in the data is obvious and it has a description in the form of a predicate $[x_i \ge b]$.

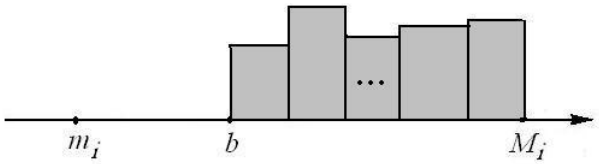

Figure 5. A regularity in the form $[x_i \ge b]$.

If we consider a one-dimensional random variable uniformly distributed on the segment $[m_i, M_i]$ with independent realizations — hits in this segment, the probability to get into $[m_i, b]$ would be geometrically estimated as

$$p = \frac{b - m_i}{M_i - m_i}, \tag{8}$$

and the probability not to get into $[m_i, b]$ l times in l independent tests – as $(1-p)^l$. This value is the probability of an event that the specified interval will be *accidentally empty* as a result of l tests. And with probability $1-(1-p)^l$ this interval will be empty not by chance, i.e. in this case, we can say that the probability of the regularity $[x_i \ge b]$ as a result of l sample tests is $1-(1-p)^l$.

To approximate the domain of admissible solutions Ω we will use a classifying tree with threshold predicates of the form $[x_i \ge b]$ at the vertices Loh (2014) constructed by precedents representing only

points-representatives of this domain Ω. The classifying tree that allocates empty segments is shown in figure 4 to illustrate the idea.

The main element of the synthesis of such a tree is the choice of threshold values b for each branch (splitting the current area G_v) and the construction of another internal or terminal vertex of the tree. The following algorithm 3 is the main procedure providing the process of synthesis of the approximation tree of the Ω domain.

Algorithm 3. Clearing up the possibility of allocating an empty segment and the selection of the predicate of the form $[x_i \geq b]$ ($[x_i \leq b]$) for the splitting of the current region G_v and building the next vertex v of the classification tree.

Input: the study area – hyperparallelepiped G_v;
Δ_1 – the minimum permitted width of the projection of the empty region.

Output: the value of the pointer $Flag$; if $Flag = 1$ then it is possible
an allocation of the empty segment and construct a conditional predicate for vertices;
if $Flag = 0$ the the area G_v cannot be split, and the instruction is issued
to build the end vertex marked with a admissible segment.

1. Select all sample points that are in the area G_v.
2. $Flag := 0$;
3. ***Cycle*** through all the coordinates of the points in the area G_v;
4: **if** $Flag := 1$ **then goto** 14;
5. Find the average distance Δ_2 between the projections of the points;
6. Find the minimum μ and maximum $\mathcal{M}$ values of the current coordinate;
7. **if** $r_1 = \mu - m_i > \Delta_1 \ \vee \ r_2 = M_i - \mathcal{M} > \Delta_1$ **then**
8: { $Flag := 1$;
9: **if** $r1 \geq r_2$ **then**
10: { $b := \mu - \Delta_2$; add a vertex with the predicate $[x_i \leq b]$ }
11: **else**
12: {$b := \mathcal{M} + \Delta_2$; add a vertex with the predicate $[x_i \geq b]$};
}
13. ***the end of the cycle***;
14. **if** $Flag := 0$ **then** add a leaf marked as a region of admissible solutions.

Each terminal vertex τ marked E_τ as "empty" region G_τ corresponds to the probability $P(E_\tau)$ which estimates a non-randomness of its detection or, in other words, the probability that the detected empty region G_τ is a regularity:

$$p(E_\tau) = 1 - (1 - p_\tau)^l, \quad p_\tau = \mathrm{V}(G_\tau)/\mathrm{V}_0, \tag{9}$$

where $\mathrm{V}(G_\tau)$ is the volume of the empty region G_τ. This volume is easy to calculate carrying out a reverse pass through the branches of the tree from the terminal vertex τ to the root of the tree "reading" all threshold predicates in the viewing vertexes passable branches and forming a description of hyperparallelepiped G_τ.

5. Numerical experiment

To illustrate the approach to the synthesis of a nonlinear neural optimization model, a sample of 17 manufacturing enterprises was used (Table 1; cost indicators are presented for the month). The

observations in the table correspond only to admissible points ($\Omega(\tilde{x}) = 1$).

Following 5 features-factors that characterize the enterprises are used:

x_1 — gross output in value terms (thousand dollars);

x_2 — cost of fixed assets (thousand dollars);

x_3 — number of employees;

x_4 — specialization ratio [§];

x_5 — marketing mix ratio [¶].

A target attribute y is a profit (thousand dollars).

Table 1. The sample for training the neural network.

Number	x_1	x_2	x_3	x_4	x_5	y
1	1404.5	345.9	170	0.313	0.061	112.5
2	1709.8	431.9	225	0.285	0.067	113.7
3	1808.7	886.2	238	0.398	0.089	193.0
4	1437.1	484.2	181	0.322	0.076	125.0
5	1496.1	724.6	177	0.367	0.085	173.4
6	1034.3	200.7	179	0.206	0.039	81.4
7	1335.0	317.6	162	0.314	0.049	106.4
8	1256.1	156.1	159	0.187	0.038	72.6
9	1581.4	364.3	255	0.319	0.054	110.7
10	1826.5	554.2	275	0.338	0.083	146.3
11	1697.7	387.7	251	0.219	0.064	112.9
12	1294.6	302.5	154	0.214	0.044	105.9
13	1174.7	483.9	215	0.324	0.075	134.5
14	1180.9	220.1	165	0.217	0.039	91.4
15	1319.0	243.6	156	0.207	0.040	98.4
16	1460.0	347.3	202	0.316	0.053	107.6
17	1478.3	313.5	186	0.211	0.044	102.3

The neural network with only one hidden layer was used (Figure6).

It is known that such networks have sufficiently great possibilities of functions approximation when training for regression Hornik et al. (1990).

[§]multiplied by 100 the ratio of the cost of finished products of the profile direction to the cost of all finished products (per month).
[¶]linear convolution of the coefficients of the main marketing indicators.

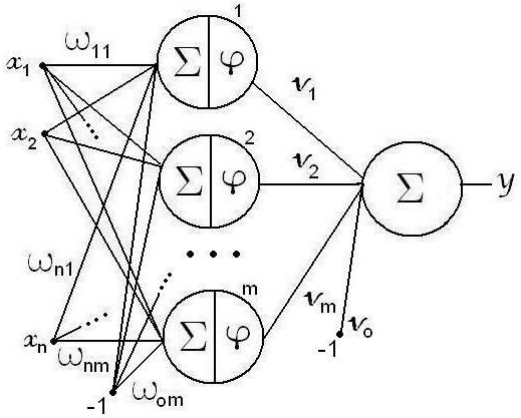

Figure 6. Neural network with only one hidden layer used in the numerical experiment.

Sigmoid activation function

$$\varphi(z) = \frac{1}{1.0 + \exp(-1.0351 \cdot z)} \tag{10}$$

was used. The error estimated at neural network training is

$$E(\tilde{w}, \tilde{V}) = \frac{1}{2} \sum_{j=1}^{l} \left(f_{NN}(X_j^{sample}) - y_j^{sample} \right)^2; \tag{11}$$

$$f_{NN}(x_1, \ldots, x_n) = -V_o + \sum_{q=1}^{m} V_q \cdot \varphi \left(-w_{0,q} + \sum_{i=1}^{n} w_{i,q} x_i \right) \tag{12}$$

is the output of the neural network at sample number j; $\tilde{w} = \{w_{i,q}, i = \overline{0,n}; q = \overline{1,m};\}$ are the input weights of the network; X_j^{sample} – j-th row of the training table; V_q, $q = \overline{0,m}$, – the output weights of the network; y_j^{sample} – profit value in the row j of the sample.

The following formulas were used:

$$\eta_k = \varphi \left(-w_{0,k} + \sum_{s=1}^{m} w_{s,k} x_s \right); \quad \frac{\partial f_{NN}}{\partial w_{i,q}} = \sum_{k=1}^{m} V_k x_i \eta_k (1 - \eta_k), \; i = \overline{1,n}; q = \overline{1,m}; \tag{13}$$

$$\frac{\partial f_{NN}}{\partial w_{0,q}} = -\sum_{k=1}^{m} V_k \eta_k (1 - \eta_k), \; q = \overline{1,m}; \quad \frac{\partial f_{NN}}{\partial V_k} = \eta_k, \; k = \overline{1,m}; \quad \frac{\partial f_{NN}}{\partial V_0} = -1; \tag{14}$$

$$\frac{\partial E(\tilde{w}, \tilde{V})}{\partial w_{i,q}} = \left(f_{NN}(X_j^{sample}) - y_j^{sample} \right) \frac{\partial f_{NN}}{\partial w_{i,q}}; \quad \frac{\partial E(\tilde{w}, \tilde{V})}{\partial V_k} = \left(f_{NN}(X_j^{sample}) - y_j^{sample} \right) \eta_k. \tag{15}$$

Well-known gradient descent formulas were used in the training:

$$w_{i,q}^t := w_{i,q}^{t-1} - \gamma_t \frac{\partial E(\tilde{w}, \tilde{V})}{\partial w_{i,q}}; \quad V_q^t := V_q^{t-1} - \gamma_t \frac{\partial E(\tilde{w}, \tilde{V})}{\partial V_k}, \tag{16}$$

where γ_t, $t = 1, 2, \ldots$ is decreasing with the growth of t learning rate.

During 94 eras of learning, there was a monotonous decrease in the average error $\frac{1}{l}\sum_{j=1}^{17}(y_j^{NN} - y_j^{sample})^2$ from 1.0878 until 0.0277.

After the neural network was trained, the problem of finding the maximum of the function realized by this neural network under the following restrictions was solved: the values of features not exceeding the boundaries of the segment

$$0,9 \min_{j=\overline{1,17}} x_{j,i}^{sample} \le x_i \le 1,1 \max_{j=\overline{1,17}} x_{j,i}^{sample} \tag{17}$$

were allowed. The following formulas were used for gradient lifting:

$$\frac{\partial f_{NN}}{\partial x_i} = \sum_{q=1}^{m} V_q \eta_q (1-\eta_q) w_{i,q}; \quad x_i^t := x_i^{t-1} + \gamma_t \Delta y^t \frac{\partial f_{NN}}{\partial x_i}, \tag{18}$$

where Δy^t is the increment at the next step of the function defined by the constructed neural network.

Exactly 250 optimization steps have been completed. As a result, the maximum $Y = \mathbf{195,54}$ was found for the values of the variables $x_1 = 1096,56$, $x_2 = 934,67$, $x_3 = 199,94$, $x_4 = 0,392$, $x_5 = 0,0371$. Note that the maximum profit value $\left(\max_{j=\overline{1,17}} y_j^{sample}\right)$ in the sample is equal to $193,0$.

Setting the optimal parameters found by the person finally making the decision will lead to an increase in profits. This is the management implications.

The scope of the model presented in the numerical example is quite wide. Such a model can be synthesized to find the values of parameters of any manufactures that maximize profits, to find the values of technical parameters of complex engineering systems. The use of a neural network makes it possible to learn, generally speaking, any nonlinear dependence that will be reflected in empirical samples.

The use of neural networks for the synthesis of target functions makes it possible to implement complex dependencies on a large number of variables, but it is difficult to find out how different values of a set of independent variables affect a particular dependent variable under certain specific conditions. Therefore sensitivity analysis should be considered an additional task. To solve it, one can use the calculation of partial derivatives by independent variables. Promising for this purpose also is the use of methods of transformation of the trained neural network into a system of logical rules, for example, represented by a decision tree Boz (2002). However, sensitivity analysis issues are beyond the scope of this article.

6. Conclusion

In conclusion, the following should be noted.

BOMD technology is an adequate tool for solving many problems of intelligent management in economics and technology. Although the results of the synthesis of optimization models from the data and decision-making using these models can give significant errors, these errors can be significantly smaller compared to the errors of subjective models since there is the consistency of decisions with real data. The objectivity of decision-making in this sense is crucial.

In this paper, we considered the synthesis of nonlinear models from data and used neural networks for this aim. A synthesis of linear models was previously studied in detail by the author in his previous

papers listed in the references. We emphasize that the use of nonlinear models is advisable only when a test of the admissibility of the application of linear models gives a negative result Donskoy (2012).

Conflict of interest

The author declares no conflicts of interest in this paper

References

Abiodun OI, Jantan A, Omolara AE, et al. (2018) State-of-the-art in artificial neural network applications: A survey. *Heliyon* 4: 1–7.

Antonova GM (2011) Application of Pattern Recognition Methods to Solve Optimization Problems Using Imitation Models. *Pattern Recognit Image Anal* 21: 113–116.

Barton D, Court D (2013) Three Keys to Building a Data-Driven Strategy. Available from: https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/three-keys-to-building-a-data-driven-strategy.

Boz O (2002) Converting A Trained Neural Network To a Decision Tree DecText - Decision Tree Extractor. *ICMLA 2002: Las Vegas, Nevada, USA* 4: 110–116.

Breiman L, Friedman JH, Olshen R, et al. (1984) *Classification and Regression Trees,* Chapman and Hal, New York, NY.

Corne D, Lones MA (2018) Evolutionary Algorithms, *In: Marti R, Panos P, and Resende M. (eds) Handbook of Heuristics*, Springer, Cham.

Donskoy VI (2018) A Synthesis of Pseudo-Boolean Empirical Models by Precedential Information. *Bulletin of the South Ural State University, Series: Mathematical Modelling, Programming and Computer Software* 11: 96–107.

Donskoy VI (2017) Extraction of Optimization Models from Data: a Decision Tree and Forest based Approach. *Taurida J Comput Sci Theory Math* 4: 59–86.

Donskoy VI (1993) Partially Defined Optimization Problems: An Approach to a Solution that is based on Pattern Recognition Theory. *J Sov Math* 65: 1664–1668.

Donskoy VI (2012) Synthesis of coordinated linear optimization models according to precedential information: an approach based on Kolmogorov complexity. *Taurida J Comput Sci Theory Math* 1: 13–25.

El-Sawi AA, Hussein MA, Zaki EM, et al. (2014) An Introduction to Genetic Algorithms: A survey. A practical Issues. *Int J Sci Eng Res* 5: 252–262.

Eremin II, Mazurov VlD (1979) *Unsteady processes of mathematical programming,* Nauka, Moscow.

Haykin S (2008) *Neural Networks and Learning Machines,* Prentice Hall.

Hornik K, Stinchcombe M, White H (1990) Univrsal Approximation of an Unknown Mapping and Derivatives Using Multilayer Feedforward Networks. *Newral Networks* 3: 551–560.

Kolmogorov AN (1991) *Algorithm, information, complexity,* Znanie, Moscow.

Loh WY (2014) Fifty Years of Classification and Regression Trees. *Int Stat Rev* 82: 329–348.

MathWorks (2017) Building Models from Data and Scientific Principles. Available from: `https://www.mathworks.com/solutions/mathematical-modeling/building-models-data-scientificprinciples.html`.

Mazurov VlD (1971) Application of Methods of Theory of Pattern Recognition in the Optimal Planning and Management. *Proceeding of I-st all-Union Conference on Optimal Planning and National Economy Management*. Moscow.

Malhotra R, Singh N, Singh Y (2011) Genetic Algorithms: Concepts, Design for Optimization of Process Controllers. *Comput Inf Sci* 4: 39–54.

Roh Y, Heo G, Whang SE (2019) A Survey on Data Collection for Machine Learning. *ArXiv: 1811.03402v2(201 [cs.LG]*.

Zhang S, Zhang C, Yang Q (2003) Data Preparation for Data Mining. *Appl Artif Intell* 17: 375–381.

AIMS Press