INTERNATIONAL DEVELOPMENT PLANNING REVIEW
ISSN:1474-6743 | E-ISSN:1478-3401

OPTIMAL CONTROL THROUGH PROBLEM SOLVING RESEARCH USING NEURAL NETWORK METHODS

Kurmanov Bakyt
MSc Student in Engineering
Kazakh-British Technical University, School of Information Technology and Engineering
Kurmanov.b@gmail.com
Orcid id-0009-0001-7660-2046

Baikonys Anuar
Senior Lecturer, Trainer
Joint-Stock Company «National Center for Advanced Studies «Orleu»
anuar_bak@mail.ru
Orcid id-0009-0006-0557-737X

Pernebayev Yerzhan
Doctoral Student
M. Auezov South Kazakhstan University
er-ji@mail.ru
Orcid id-0009-0007-7473-8867

Kurmangaliyeva Nurgul
PhD in Computer Science,
Alikhan Bokeikhan University
nurgulkk62@mail.ru
Orcid id-0000-0003-1709-662X

Ospanova Dinara
Head of the Department
Shakarim State University Of Semey
ODM-1778@mail.ru
Orcid id-0000-0002-6131-4113

Alkhanova Gulnur
Head of the Career Department
Alikhan Bokeikhan University
gulnur_alhanova@mail.ru

Orcid id-0000-0002-1151-7254

Abstract

In this study, we discuss methods for solving optimal control problems based on neural network methods. We are studying a hierarchical dynamic two-level surface water quality control system. The system consists of a supervisory authority (government) and several agents (enterprises). We consider this problem from the point of view of agents. In this case, we solve the optimal control problem with constraints. To solve this problem, we use the Pontryagin maximum principle, by which we obtain optimality conditions. To solve the emerging ODES, we use a live broadcast neural network. We provide an overview of existing methods for studying such problems and an overview of neural network training methods. To estimate the error of the numerical solution, we propose to use a defect analysis method adapted for neural networks. This allows us to obtain quantitative estimates of the error of the numerical solution. We give examples of using our method to solve a synthetic problem and a model of surface water quality control. We compare the results of these examples with the known solution (when it is provided) and the results of the survey method. In all cases, the errors estimated by our method are of the same order as the errors compared to the known solution. Moreover, we study the problem of surface water quality control when other methods do not provide a solution. This is due to the relatively large time interval and/or the case of multiple agents. In the latter case, we are looking for a Nash equilibrium between the agents. Thus, in this study we show the ability of neural networks solve various problems, including optimal control problems and differential games, and we show the ability to quantify the error. From the numerical results, we conclude that the presence of a supervisor is necessary to achieve sustainable development.

Keywords

Optimal control, differential games, neural network, Nash equilibrium, Pontryagin’s maximum principle

Introduction

Optimal control issues arise in the study of control systems of different nature: mechanical, economic, ecological-economic, and social. Any problem of optimal control implies the presence of some goal, which is to be achieved in an optimal way, taking into account the constraints on controls and on the differential equation describing the state of the system. A significant number of works by various authors are devoted to the solution of such problems; in particular, an overview of solution methods is given in [Rao, 2009]. There are two approaches to solving such problems: indirect and direct.

An indirect approach is to study the system by analytical methods of variational calculus or dynamic programming, or using Pontryagin's maximum principle [Rao, 2009]. These methods are used to analytically derive optimality conditions. In the case of Pontryagin's maximum principle, the problem is reduced to a two-point boundary value problem and the problem of maximizing the Hamiltonian. Two-point boundary value problems for systems of differential equations are rarely solved analytically and require the use of numerical methods, such as the shooting method [Rao, 2009]. In general, the matter of solving two-point boundary value problems is topical [Hua, 1992; Sung, 2001; Holsapple, Venkataraman, 2004], and methods for solving it are still proposed today [Malkin, 2016], some of them are based on the use of neural networks [Hua, 1992].

It should be noted that the shooting method is not without a number of significant drawbacks; namely, there is a problem of convergence over long time intervals [Rao, 2009], which the authors also faced earlier [Reshitko, Ugolnitsky, Usov, 2020].

In the direct approach [Rao, 2009], the state function of the system, whose change is described by a system of differential equations, and unknown optimal controls are approximated in some way (often by different polynomials). In this case, unknown functions are transformed into functions of a known form, but with unknown coefficients, and then the scalar optimization problem is solved. It is important to note that the main disadvantage of direct methods is the need to transition to the problem of numerical scalar optimization, where the global optimality of the solution is not guaranteed.

The task of developing new methods for studying the problem of optimal control remains relevant today. In this paper, it is proposed to use neural networks to solve optimal control problems, following [Gorbachenko, Artyukhina, 2007; Kovalenko, Chernomorets, Petina, 2017; Yize, Yuanyuan, Baisen, 2019; Cybenko, 1989; Zhou et al., 2017; Nazimi and Karami, 2017; Andreeva, Tsiruleva, 2018; Pirogov S.P.; Ustinov N.N.; Smolin N.I., 2018; Surinskiy, D.O., Savchuk, I.V., Solomin, E.V., Kovalyov, A.A., 2019]. The main idea of the method described below is that, first, the optimal control problem is studied by analytical methods and optimality conditions are defined. Then the unknown functions are presented as a neural network that is trained to meet the optimality conditions. Thus, indirect and direct approaches are combined. The use of neural networks is justified by the fact that neural networks are universal approximators, i.e., they can theoretically approximate any solution, and besides, the process of solving the problem is parallelized on video cards. A similar approach to the study of optimal control problems was used in [Nazemi, Karami, 2017; Andreeva and Tsiruleva, 2018]. In contrast to [Nazemi, Karami, 2017], a single neural network is used to approximate all unknown functions in this work; in addition, the proposed approach is used not only for the optimal control problem, but also for solving a noncooperative differential game of several agents. A method adapted for neural networks for assessing the accuracy of the obtained solution based on the analysis of the solution defect is also proposed. This method allows quantification of the accuracy of the solution obtained by the neural network method. In this case, the boundary conditions are satisfied accurately, and the solution itself is continuous and differentiable. Moreover, in the considered examples, the neural network approach made it possible to obtain solutions over longer time intervals than the shooting method. The use of neural networks is explained by the development of network architectures, numerical methods for their training [Adam: a method for stochastic optimization; Cyclical learning rates. . .; Lookahead optimizer. . .] and software systems [PyTorch; autograd; jax; Gym], which allows adapting a neural network to solve almost any problem [Cybenko, 1989; Zhou et al., 2017].

The article is organized as follows: the “Problem Statement” section describes the problem of surface water quality control and considers the proposed method for studying optimal control problems on its example. Further, the “Research Methods” section provides an algorithm for solving the problem using neural networks, various network training methods, and a method for assessing the error. The “Examples of calculations” section contains examples of the method application on a model example and the problem of surface water quality control.

Problem Statement

The neural network approach is considered on the example of the problem of surface water quality control, which was already studied by the authors in [Reshitko, Ugolnitsky, Usov, 2020]. The problem is stated as follows.

The activity of enterprises that discharge pollutants into water as a result of their activity is considered. The possibility of controlling the discharge of pollutants by the enterprise on the part of the regulatory authority by setting the fee for the discharge of pollutants is investigated. At the same time, it is important for enterprises to maintain production volumes. The goals of reducing the discharge of pollutants and maintaining production volumes are, in a sense, opposite, since the increase in production volumes leads to the increase in the discharge of pollutants, therefore the goal is to limit the concentration of pollutants in the water body within acceptable values. The problem is considered in a dynamic setting, i.e., the discharge of pollutants is controlled at a given time interval.

Thus, the river water quality control system under consideration includes a regulatory body (supervisor) and several industrial enterprises (agents). As a result of production, industrial enterprises produce pollutants, which, together with wastewater, are discharged into the watercourse. The problem is considered in a game setting and has a hierarchical structure. The supervisor goes first, assigning the amount of fees to the enterprises for the discharge of pollutants. Enterprises determine the degree of wastewater treatment when the decision of the supervisor is already known. The paper studies the case of an indifferent supervisor. In this case, the pollution discharge fee is considered a given function, and the model becomes one-level. Agents tend to maximize their target functionals by determining the degree of wastewater treatment:

$$\max_{u_i} \int_{t_0}^{t_1} (1 - \eta_i) F_i(t, \phi_i, \rho, u) dt. \quad (1)$$

The constraints on agent controls are

$$u_i \in \left[u_i^{min}, u_i^{max}\right]. \quad (2)$$

The following system dynamics equations are taken:

$$\dot{\phi} = f(t,\ \phi_i, \rho, u), \qquad \phi(t_0) = \phi_0, \tag{3}$$

$$\dot{\rho} = g(t,\ \phi_i, \rho, u), \qquad \rho(t_0) = \rho_0, \tag{4}$$

$$\phi(t)\colon \mathbb{R} \to \mathbb{R}^N,$$

$$\rho(t)\colon \mathbb{R} \to \mathbb{R}.$$

Here the payoff function of the *i*-th agent is as follows:

$$F_i(t,\ \phi_i, \rho, u) = z_i R_i(\phi_i) - W_i(\phi_i) C_i(u_i) - \frac{\rho v_i W_i(\phi_i)(1-u_i)}{\rho_{max}}.$$

Here the term $z_i R_i(\phi_i)$ denotes the agent's profit from the sale of manufactured products, $W_i(\phi_i)C_i(u_i)$ is the agent's costs for the treatment of the discharged pollutants, $\frac{\rho v_i W_i(\phi_i)(1-u_i)}{\rho_{max}}$ is the agent's fee for the discharge of untreated pollutants.

The functions describing the system dynamics are

$$f_i(t,\ \phi_i, \rho, u) = -\beta_i \phi_i + \eta_i F_i(t,\ \phi_i, \rho, u),$$

$$g(t, \phi, \rho, u) = -k\rho + \sum_{i=1}^{N} W_i(\phi_i)(1-u_i).$$

Here $\beta_i \phi_i$ characterizes the obsolescence of production assets, and $\eta_i F_i(t,\ \phi_i, \rho, u)$ is the reinvestment of part of the agent's profit in the development of the enterprise. $k\rho$ sets the self-purification of the water body over time, and $\sum_{i=1}^{N} W_i(\phi_i)(1-u_i)$ is the total discharge of untreated pollutants by agents.

The following designations are accepted: $u = (u_1, u_2, \dots, u_N)$; u_i is the degree of wastewater treatment by the *i*-th agent; $[t_0, t_1]$ is the time interval under consideration; N is the number of agents; $\rho(t)$ is the concentration of pollutants, whose change over time is described by equation (3);
$\phi = (\phi_1, \phi_2, \dots, \phi_N)$ is the value of agent's production assets, the assets of the *i*-th agent are described by equation (4); $R_i(\phi_i)$ is the production function of the *i*-th agent; $W_i(\phi_i)$ is the volume of pollutants discharged by the i-th agent; $\rho_{max}$is the maximum allowable concentration of pollutants; $v_i(t)$ is the amount of the fee for the discharge of a unit of pollutants for the *i*-th agent; u_i^{min}, u_i^{max} is the interval of possible control values of the agent u_i; $\eta_i$is the share of profit reinvested by the agent in the development of the enterprise; z_i is the agent's profit per unit sold; $C_i(u_i)$is the agent's costs for wastewater treatment; *k* is the coefficient of decay of pollutants; $\rho_0$is the concentration of pollutants at the initial time; $\phi_0 = (\phi_{01}, \phi_{02}, \dots, \phi_{0N})$ is the value of agent's production assets at the initial time.

The study is conducted in a game setting, it is assumed that all agents make their decisions independently of each other, striving to maximize their target functionals. As a result, the Nash equilibrium is established in the game of agents. Moreover, each agent solves the optimal control problem (1)-(4). To solve it, the Pontryagin's maximum principle is used in [Rao, 2009]. The Hamiltonians H_i are derived:

$$H_i(t,\ \phi_i, \rho, \lambda_{i1}, \lambda_{i2}, u) = (1-\eta_i)\left(z_i R_i(\phi_i) - W_i(\phi_i)C_i(u_i) - \frac{\rho v_i W_i(\phi_i)(1-u_i)}{\rho_{max}}\right) +$$
$$+\sum_{j=1}^{N}\lambda_{ij}\left(-\beta_i\phi_i + \eta_i\left(z_i R_i(\phi_i) - W_i(\phi_i)C_i(u_i) - \frac{\rho v_i W_i(\phi_i)(1-u_i)}{\rho_{max}}\right)\right) +$$
$$+\lambda_{i0}\left(-k\rho + \sum_{i=1}^{N} W_i(\phi_i)(1-u_i)\right)$$

and the equations for the introduced variables $\lambda_i(t)$ are written out:

$$\dot{\lambda}_{ii} = -\frac{dH_i}{d\phi_i} = -\left(\left(\lambda_{i0}\frac{\partial W_i}{\partial \phi_i}(1-u_i)\right) - \beta_i\lambda_{ii} +\right.$$

$$+(1-\eta_i+\eta_i\lambda_{ii})\left(z_i\frac{\partial R_i}{\partial \phi_i}-\frac{\partial W_i}{\partial \phi_i}C_i(u_i)-\frac{\rho v_i\frac{\partial W_i}{\partial \phi_i}(1-u_i)}{\rho_{max}}\right)\Bigg), \qquad \lambda_{ii}(t_1)=0,$$

$$\dot{\lambda}_{ij}=-\frac{dH_i}{d\phi_j}=-\Bigg(\left(\lambda_{i0}\frac{\partial W_j}{\partial \phi_j}(1-u_j)\right)-\beta_i\lambda_{ij}+$$

$$+\eta_j\lambda_{ij}\left(z_j\frac{\partial R_j}{\partial \phi_j}-\frac{\partial W_j}{\partial \phi_j}C_j(u_j)-\frac{\rho v_j\frac{\partial W_j}{\partial \phi_j}(1-u_j)}{\rho_{max}}\right)\Bigg), \qquad \lambda_{ij}(t_1)=0, \quad i\neq j,$$

$$\dot{\lambda}_{i0}=-\frac{dH_i}{d\rho}=-\Bigg(-\lambda_{i0}l-(1-\eta_i)\frac{v_iW_i(\phi_i)(1-u_i)}{\rho_{max}}$$
$$-\sum_{j=1}^{N}\lambda_{ij}\eta_j\frac{v_jW_j(\phi_j)(1-u_j)}{\rho_{max}}\Bigg), \ \lambda_{i0}(t_1)=0.$$

As a result, two subproblems are obtained (i = 1, ..., N):

1) a problem of maximizing the Hamiltonians:

$$\max_{u_i} H_i\,; \tag{5}$$

2) a two-point boundary value problem for state variables and conjugate variables:

$$\dot{\phi}_i=-\beta_i\phi_i+\eta_iF_i(t,\,\phi_i,\rho,u), \qquad \phi_i(t_0)=\,\phi_{0i}, \tag{6}$$

$$\dot{\rho}=-k\rho+\sum_{i=1}^{N}W_i(\phi_i)(1-u_i)\,, \quad \rho(t_0)=\,\rho_0, \tag{7}$$

$$\dot{\lambda}_{ii}=-\frac{dH_i}{d\phi_i}, \qquad \lambda_{ii}(t_1)=0,$$

$$\dot{\lambda}_{ij}=-\frac{dH_i}{d\phi_j}, \qquad \lambda_{ij}(t_1)=0, \quad i\neq j, \tag{8}$$

$$\dot{\lambda}_{i0} = -\frac{dH_i}{d\rho}, \qquad \lambda_{i0}(t_1) = 0. \tag{9}$$

Problems (5)-(9) are solved in two stages. The first stage is to solve problem (5), which for N = 1 is the problem of maximizing the Hamiltonian, and for N > 1, it is the problem of reaching the Nash equilibrium between all agents. Thus, the optimal controls of agents u_i are determined with respect to spatial and introduced variables $u_i = u_i(t,\ \phi_i, \rho,\ \lambda_{i1}, \lambda_{i2})$. For a particular form of input functions, problem (5) is solved analytically, and in the general case, optimality conditions of the following form are obtained from (5):

$$\frac{\partial H_i}{\partial u_i} = -\lambda_{i0} W_i(\phi_i) + (1 - \eta_i + \eta_i \lambda_{ii}) \left(-W_i(\phi_i) \frac{\partial C_i}{\partial u_i} + \frac{\rho v_i W_i(\phi_i)}{\rho_{max}} \right) = 0,$$

$$\frac{\partial^2 H_i}{\partial u_i^2} = -(1 - \eta_i + \eta_i \lambda_{ii}) W_i(\phi_i) \frac{\partial^2 C_i}{\partial u_i^2} < 0,$$

$$u_i \in \left[u_i^{min}, u_i^{max}\right].$$

The second stage is to solve the two-point boundary value problem (6)-(9) using the found functions $u_i(t,\ \phi_i, \rho,\ \lambda)$ or the optimality conditions for (5).

Research Methods

To study the problem of surface water quality control (5)-(9), it is proposed to use an approach based on a neural network, hereinafter referred to as a neural network approach. The essence of the approach lies in the fact that the desired solution is represented as a neural network $NN : \mathbb{R} \to \mathbb{R}^M$, where M is the number of unknown functions. The input of the network is the time coordinate *t*, and the output of the network of length *M* includes the values of the state variables ϕ, ρ, introduced variables λ_i, and, possibly, the controls *u*. The latter depends on whether an analytical solution of problem (5) is possible. Such a structure of the neural network corresponds to the desired solution of the problem, since, in the end, functions depending only on time are sought. The application of the approach is preceded by an analytical study of the system (5)-(9), which can result, for example, in solving problem (5) or obtaining optimality conditions for (5), which will simplify the application of the method.

The algorithm for solving problem (5)-(9) using the neural network approach consists in the following:

1) determining the configuration of the neural network *NN(t, w)* for a specific set of input data
bearing in mind that the input of the network is the time coordinate, and its output includes the values of the state variables ϕ, ρ, introduced variables λ, and other functions that were not determined during the analytical study;

2) introducing a loss function that describes the deviation of the network values from the optimality conditions (5)-(9);

3) choosing the network training method;

4) generating a training sample;

5) optimizing the loss function by the chosen method; training stops when a sufficiently small value of the loss function is reached;

6) evaluating the obtained results, turning to point 1, if necessary.

Let us explain the above points of the algorithm.

The neural network introduced at the first stage has a fixed structure for the problem and is described by its own weights $w \in \mathbb{R}$. The value of the solution at time *t* is determined by the pair *(t, w)*. The problem, therefore, is to find the weights *w* so that the neural network satisfies conditions of the problem (5)-(9) with some required accuracy. This approach is similar to the collocation method [Rao, 2009], where the solution is sought in the form of polynomials of various 2 with unknown coefficients. To solve the problem, we take a set of points $T \in [t_0, t_1]$ and require the conditions (5)-(9) to be satisfied at them for the neural network *NN (i = 1..., N, t* $\in T$*)*.

The maximum condition for the Hamiltonians (5) takes the form

$$\max_{u_i} H_i\,(t, NN(t,w), u^*). \tag{10}$$

A part of the output of the neural network (or its derivative $\frac{\partial NN}{\partial t}(t,w)[x]$), corresponding to variable *x*, is denoted by $NN(t,w)[x]$. The two-point boundary value problem (6)-(9) takes the form

$$\frac{\partial NN}{\partial t}[\phi_i] = f_i(t, NN(t,w), u^*), \qquad NN(t,w)[\phi_i](t_0,w) = \phi_{0i}, \tag{11}$$

$$\frac{\partial NN}{\partial t}[\rho] = g(t, NN(t,w), u^*), \qquad NN(t,w)[\rho](t_0,w) = \rho_0 \tag{12}$$

$$\frac{\partial NN}{\partial t}[\lambda_{ii}] = -\frac{dH_i}{d\phi_i}(t, NN(t,w), u^*), \qquad NN(t,w)[\lambda_{ii}](t_1, w) = 0,$$

$$\frac{\partial NN}{\partial t}\left[\lambda_{ij}\right] = -\frac{dH_i}{d\phi_j}(t, NN(t,w), u^*), \qquad NN(t,w)\left[\lambda_{ij}\right](t_1, w) = 0, \qquad i \neq j, \qquad (13)$$

$$\frac{\partial NN}{\partial t}[\lambda_{i0}] = -\frac{dH_i}{d\rho}(t, NN(t,w), u^*), \qquad NN(t,w)[\lambda_{i0}](t_1, w) = 0. \qquad (14)$$

Here u^* denotes either the corresponding output of the network $NN(t,w)[u]$ or the analytically found optimal controls of agents $u^*\left(t, NN(t,w)\right)$. It should be noted that the boundary conditions for equations (11)-(14) can be satisfied by changing the configuration of the neural network. It suffices to look for a solution in the form [Yadav, Yadav, Kumar, 2015] $\overline{NN}(t,w) = A(t) + F\left(t, NN(t,w)\right)$, where the first term satisfies the boundary condition, and the second term is zero at the appropriate moment in time. Now the function $\overline{NN}(t,w)$ replaces the neural network and automatically satisfies the boundary conditions. To simplify calculations, $\overline{NN}(t,w)$ will be further denoted as $NN(t,w)$.

Feedforward neural networks that sequentially pass the input signal through a sequence of layers are used in the study. The examples in this article use networks with three hidden layers. Each layer performs a linear transformation of the input signal and applies an activation function to it. Thus, the *i*-th layer has the form

$$l_i(x_{i-1}) = s_i(x_{i-1}w_i + b_i) \to x_i,$$

where w_i, b_i are the weights of the *i*-th layer, s_i is the activation function, x_{i-1} is the signal of the previous layer or the input signal, w_i is the matrix $N_i \times M_i$, b_i is the length vector of M_i, N_i is the length of the input vector, M_i is the length of the output layer. Parameters M_i, N_i are chosen by the researcher taking into account the dimensions of neighboring layers, the size of the input and the required size of the output. A neural network with *k* layers has the following form:

$$s_k(s_{k-1}(\dots s_2(s_1(x_0w_1 + b_1)w_2 + b_2) \dots w_{k-1} + b_{k-1})w_k + b_k) \to x_k.$$

At the second stage, the loss function *L* is determined, which describes the deviation of the neural network output from conditions (10)-(14):

$$L = \frac{1}{|T|}\sum_{t\in T}\sum_{i=1}^{N}\left(\left(\frac{\partial NN}{\partial t}[\phi_i] - \ f_i(\cdot)\right)^2 + \left(\frac{\partial NN}{\partial t}[\rho] - \ g(\cdot)\right)^2 + \left(\frac{\partial NN}{\partial t}[\lambda_{ii}] - \ \frac{dH_i}{d\phi_i}(\cdot)\right)^2 +$$

$$+\left(\frac{\partial NN}{\partial t}[\lambda_{ij}] - \frac{dH_i}{d\phi_j}(\cdot)\right)^2 + \left(\frac{\partial NN}{\partial t}[\lambda_{i0}] - \frac{dH_i}{d\rho}(\cdot)\right)^2 + U^2\Bigg). \qquad (15)$$

Here U denotes the optimality condition for controls, which is added if the optimal control is not found analytically. The condition depends on the specific problem, an example is the first-order maximum condition $U = \frac{\partial H}{\partial u}(t, NN(t, w))$.

At the third stage, it is possible to choose various optimization methods for network training. Normally, modifications of stochastic gradient descent are used for neural networks, such as momentum, Nesterov's momentum [Sutskever et al., 2013], Adam [Adam: a method for stochastic optimization], Lookahead [Lookahead optimizer. . .] which are aimed at accelerating convergence. The choice of learning rate (lr) for these algorithms is also important. One of the methods for choosing it is the heuristic cyclicLr [Cyclical learning rates. . .], which avoids getting stuck in local extrema. Its idea is to cyclically change lr within certain limits with a gradual decrease and a shift of the limits to zero. The examples below use the adam gradient descent modification and the cyclicLr strategy for choosing learning rate.

At the fourth stage, a training sample is created. Since the solution of the problem is reduced to minimizing the loss function (15) introduced above, then the training sample also consists of the set of points $T \in [t_0, t_1]$ used above, at which conditions (11)-(14) must be satisfied. Such a sample structure is possible because the problem parameters (including spatial and introduced variables) are already reflected in the loss function. The researcher chooses the number of points and their values. In the examples below, the training sample is taken uniformly from the considered interval $[t_0, t_1]$ and is additionally normalized to the interval $[-0.5, 0.5]$. To check the solution, 2 times more points are used than in the training sample.

At the fifth stage, the neural network is trained by the chosen method, the $\min_{w} L$ problem is solved. This problem requires to find the derivative of the neural network, both by the input t and by the weights w_i, b_i. Given that each layer performs a linear transformation of the input, and the derivative of the activation function is known, then, using the formula for the derivative of a complex function, it is relatively easy to calculate the derivatives $\frac{\partial NN}{\partial t}$ and $\frac{\partial NN}{\partial w}$. However, most libraries for machine learning (e.g., PyTorch, Tensorflow) perform this operation automatically. When calculating on a video card, it is expedient to process several examples from the sample at once to speed up the calculations. Modern libraries for machine learning allow writing out the loss function in vector form, which processes any number of examples from the sample. In this case, at each step, the value of L is minimized over the selected subset of examples. Adding random noise to the sample during training will increase the sample size and improve the convergence of the chosen method.

The last step is to evaluate the result. It is traditional for machine learning problems to estimate the value of the loss function, since its smallness indicates (in this case) that the trained network satisfies conditions (10)-(14) with some accuracy. However, such a method does not quantify this satisfaction. Therefore, the paper proposes using the value of the loss function at the training stage and then evaluating the obtained solution more accurately. It is proposed to use methods based on the analysis of the solution defect [Auzinger, 2011; Stetter, 1978; Zadunaisky, 1976]. Here is an approach to estimating the error for the case when the optimal controls are known, i.e., the system of differential equations (6)-(9) is solved. First, a solution to the original problem is found by the method described, i.e., such weights w_1 that provide a small value of L are sought. Next, the quantity

$$D = \frac{\partial NN}{\partial t}(\mathrm{t}, w_1) - f\big(t, NN(t, w_1)\big)$$

is introduced, which is called a defect, where f denotes the right-hand sides of equations (11)-(14); and an auxiliary problem is considered:

$$\dot{z} = f(t, z) + D(t), \qquad z(t^*) = NN(t^*, w_1). \tag{16}$$

Here t^* corresponds to one of the ends of the interval $[t_0, t_1]$ in accordance with the boundary conditions in (11)-(14). It should be noted that $NN(t, w_1)$ is an exact solution to the auxiliary problem (16). Now, a solution to problem (16) is found by the described method and the weights w_2 are obtained. Further, the error of the original solution is estimated as $e(t) = NN(t, w_2) - NN(t, w_1),\ t \in T$. If unsatisfactory error values are obtained, the researcher can change the network structure (increase the number and size of layers, change the activation functions), the network training method, the size and type of the training sample, and then repeat the network training process. In particular, an iterative increase in the size and number of layers of the neural network is possible due to the proposed error estimation method.

Examples of calculations

The source code for all examples is available on github [NN-for-Optimal-Control]. The easiest way to run the examples is to use google colab. Examples 1 and 2 are model ones and demonstrate the operation of the neural network method and the assessment of its accuracy in comparison with other methods. chemical methods.

Example 1

This example considers an optimal control problem that can be solved analytically. For this problem, the proposed solution method is used, which makes it possible to compare the error of the method with respect to the exact solution. An optimal control problem of the following form is considered [Kamien, Schwartz, 1991]:

$$\max_{u(t)} \int_1^5 (ux - u^2 - x^2)dt,$$

$$\dot{x} = x + u, \qquad x(1) = 2.$$

Here *x* is a spatial variable and *u* is an unknown control. There are no restrictions on control. The optimal control is found using Pontryagin's maximum principle:

$$H = ux - u^2 - x^2 + \lambda(x + u),$$

$$\dot{\lambda} = -H_x = -u + 2x - \lambda, \qquad x(1) = 2, \qquad \lambda(5) = 0,$$

$$H_u = x - 2u + \lambda,$$

$$H_{uu} = -2,$$

$$x - 2u + \lambda = 0 \to u^* = \frac{x + \lambda}{2}.$$

The exact solution of the two-point boundary value problem has the form

$$x(t) = C_1 e^{\sqrt{3}t}\left(1 + \frac{2\sqrt{3}}{3}\right) + C_2 e^{-\sqrt{3}t}\left(1 - \frac{2\sqrt{3}}{3}\right),$$

$$\lambda(t) = C_1 e^{\sqrt{3}t} + C_2 e^{-\sqrt{3}t},$$

$$C_1 = \frac{6e^{-\sqrt{3}}}{2\sqrt{3} + 3 - 3e^{8\sqrt{3}} + 2\sqrt{3}e^{8\sqrt{3}}},$$

$$C_2 = \frac{6e^{9\sqrt{3}}}{2\sqrt{3} + 3 - 3e^{8\sqrt{3}} + 2\sqrt{3}e^{8\sqrt{3}}}.$$

Now, the solution is found by the neural network method according to the formulated algorithm.

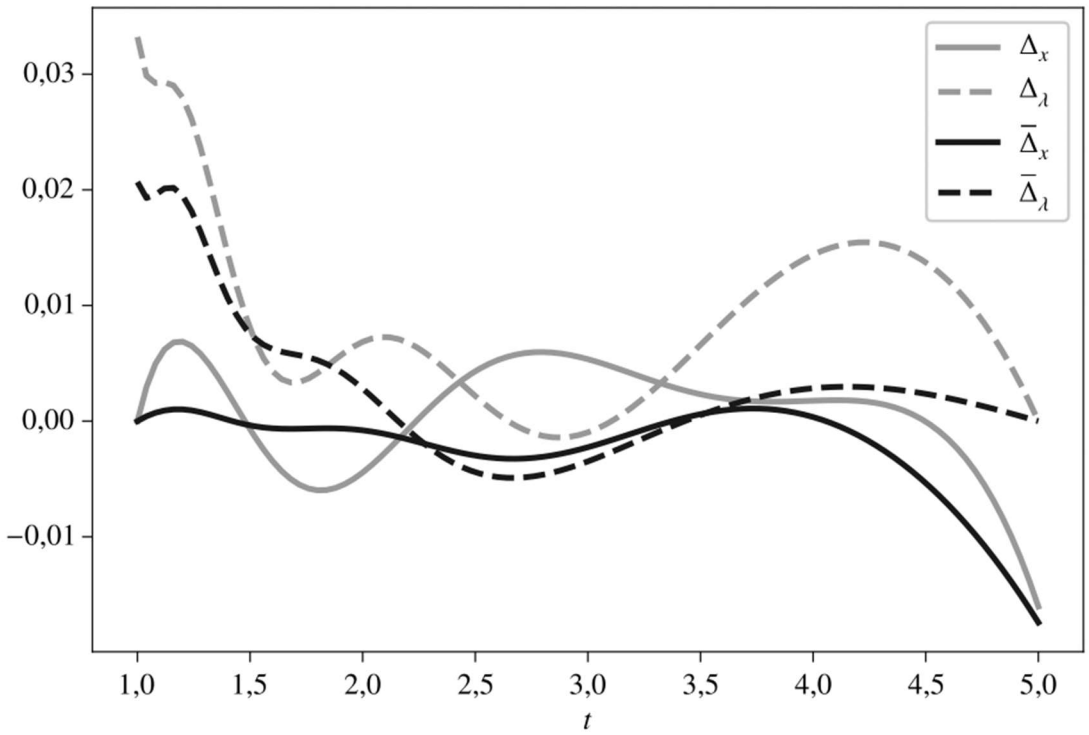
A neural network of three hidden layers with sigmoid activation functions and linear activation on the output layer is introduced. It is denoted as *nn(t)*, and its outputs as $nn_x(t), nn_l(t)$, respectively. The input of the network corresponds to the time, which is normalized as $t = \frac{t-1}{4} - 0.5$; thus, *t* lies in the interval $[-0.5;\ 0.5]$. The number of neurons in the layers is 4, 8 and 32, respectively, counting from the input of the network. To satisfy the boundary conditions, a solution is sought in the following form: $\overline{nn}(t) = A(t) + F\big(t, nn(t)\big)$, where $A(t) = [1, 0]$, $F(t) = [(t + 0.5)nn_x(t), (t - 0.5)nn_y(t)]$. A loss function that describes the deviation of the neural network from the condition of the problem is introduced:

$$L = \left(\frac{\partial \overline{nn}}{\partial x}[x] - (\overline{nn_x} + u^*)\right)^2 + \left(\frac{\partial \overline{nn}}{\partial \lambda}[\lambda] - (-u + 2\overline{nn_x} - \overline{nn_\lambda})\right)^2.$$

As a training sample, a set of time points *T* uniformly distributed on the interval [1, 5] is introduced. The example takes 51 points, including the ends of the interval. Thus, the problem of minimizing the average value of *L* is solved:

$$\min_w \overline{L(t)}, \quad t \in T.$$

To solve it, a modification of the adam gradient descent method with cyclicLr strategy for choosing learning rate is used. As a result of training, the average value of L decreased to 0.001. The graphs below compare the obtained trajectories with the exact solution. The error with respect to the exact solution, $\Delta = \max_{t \in T}(|nn_1(t) - [x, \lambda]|)$, and the error based on the value of the solution defect, $\overline{\Delta} = \max_{t \in T}(|nn_2(t) - nn_1(t)|)$, are also estimated. Here $nn_1(t)$ is the solution to the problem by the neural network method, $nn_2(t)$ is the solution of an auxiliary problem for which the exact solution is $nn_1(t)$, obtained by the neural network method. The values obtained are $\Delta = 0.016;\ 0.033$ and



$\overline{\Delta} = 0.017;\ 0.02$ for $x(t), \lambda(t)$, respectively.

Fig. 1. Comparison of the solution error with respect to the exact solution Δ_x, Δ_λ and the solution error calculated by the proposed method, $\overline{\Delta}_x, \overline{\Delta}_\lambda$, for model example 1

Since the exact solution is known for the problem, the neural network method is compared with the exact solution, and the error of the solution is also estimated. Figure 1 shows that the values of Δ and $\overline{\Delta}$ are of the same order. Thus, the possibilities of the proposed method are demonstrated on a model problem.

Example 2

This example considers the previously formulated problem of surface water quality control (1)-(4) for the case of a single agent. An exact analytical solution to it has not been found. The results of applying the proposed neural network method are compared with the results obtained using the shooting method implemented by the methods of the library for Python SciPy [SciPy]. In this case, problem (1)-(4) takes the form

$$\max_{u\in[u_{min},u_{max}]} \int_{t_0}^{t_1} F(t,\phi,\rho,u)dt,$$

$$\phi = -\beta\phi + \eta F(t,\phi,\rho,u),$$

$$\rho = -k\rho + W(\phi)(1-u),$$

$$F(t,\phi,\rho,u) = (1-\eta)\left(zR(\phi) - W(\phi)C(u) - \frac{\rho v W(\phi)(1-u)}{\rho_{max}}\right).$$

The calculations were carried out for the case of $N = 1$, $t_0 = 0$, $t_1 = 1$ month, $\beta = 0.5$, $\eta = 0.5$, $k = 0.5$, $z = 0.1$, $u_{max} = 0.9$, $u_{min} = 0$, $\rho_{max} = 1$ mg/l, $\phi_0 = 1$ c. u., $\rho_0 = 0$ mg/l, $v(t) = 1$, $R(\phi) = \phi$, $W(\phi) = \phi$, $C(u) = u^2$. The Hamiltonian has the form

$$H(\cdot) = \lambda_0\big(-k\rho + w(\phi)(1-u)\big) - \beta\phi\lambda_1 + (1-\eta+\eta\,\lambda_1)\left(zR(\phi) - W(\phi)C(u) - \frac{\rho v W(\phi)(1-u)}{\rho_{max}}\right),$$

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial \phi}, \qquad \lambda_1(t_1) = 0,$$

$$\dot{\lambda}_0 = -\frac{\partial H}{\partial \rho}, \qquad \lambda_0(t_1) = 0.$$

The maximum of the function H with respect to u is found. To do this, the zero of its derivative is found and, taking into account restrictions on u, the following is obtained:

$$u^{opt} = \begin{cases} \text{argmax}\big(H(u_{min}), H(u_{max}), H(u^*)\big), & u^* \in [u_{min}, u_{max}], \\ \text{argmax}\big(H(u_{min}), H(u_{max})\big), & u^* \notin [u_{min}, u_{max}], \end{cases}$$

where

$$u^* = \frac{-\lambda_0 \rho_{max} + \rho v(1 - \eta + \eta\,\lambda_1)}{2\rho_{max}(1 - \eta + \eta\,\lambda_1)}$$

is the zero of the derivative of *H* with respect to *u.*

Considering the found control, the problem is solved by the neural network method.

As in Example 1, a network of three layers with sigmoid activation functions and linear activation on the output layer is introduced. It is denoted as $nn(t)$, and its outputs as $nn_\phi(t)$, $nn_\rho(t)$, $nn_{\lambda_1}(t)$, $nn_{\lambda_0}(t)$, respectively. The input of the network corresponds to the time, which is normalized as $t = t - 0.5$; thus, *t* lies in the interval $[-0.5;\ 0.5]$. The number of neurons in the layers is 4, 16 and 64, respectively, counting from the input of the network. To satisfy the boundary conditions, a solution is sought in the following form: $\overline{nn}(t) = A(t) + F\big(t, nn(t)\big)$, where $A(t) = [\phi_0, \rho_0, 0, 0]$, $F(t) = [(t + 0.5)nn_\phi(t), (t + 0.5)nn_\rho(t), (t - 0.5)nn_{\lambda_1}(t), (t - 0.5)nn_{\lambda_0}(t)]$. A loss function that describes the deviation of the derivative of the neural network from the condition of the problem is introduced:

$$L = \left(\frac{\partial \overline{nn}}{\partial \phi}[\phi] - \dot{\phi}(\overline{nn}(t), u^{opt})\right)^2 + \left(\frac{\partial \overline{nn}(t)}{\partial \rho}[\rho] - \dot{\rho}(\overline{nn}, u^{opt})\right)^2 +$$

$$+ \left(\frac{\partial \overline{nn}(t)}{\partial \lambda_1}[\lambda_1] - \dot{\lambda}_1(\overline{nn}, u^{opt})\right)^2 + \left(\frac{\partial \overline{nn}(t)}{\partial \lambda_0}[\lambda_0] - \dot{\lambda}_0(\overline{nn}, u^{opt})\right)^2.$$

Here $\dot{\phi}, \dot{\rho}, \dot{\lambda}_1, \dot{\lambda}_0$ denote the right-hand sides of the corresponding equations. As a training sample, a set of time points *T* chosen by a uniform partition of the interval [0, 1] is introduced. The example takes 1000 points, including the ends of the interval. Thus, the problem of minimizing the average value of *L* is solved:

$$\min_w \overline{L(t)}, \quad t \in T.$$

To solve it, a modification of the adam gradient descent method with cyclicLr strategy for choosing learningrate is used. As a result of training, the average value of L decreased to 10^{-6}. Error values are obtained $\overline{\Delta} = 10^{-5}; 0.0001; 0.0001; 0.0001$ for $\phi(t), \rho(t), \lambda_1(t), \lambda_0(t)$, respectively. Figures 2 and 3 compare the results with the solution obtained by the shooting method. The solutions obtained by the two methods are identical.

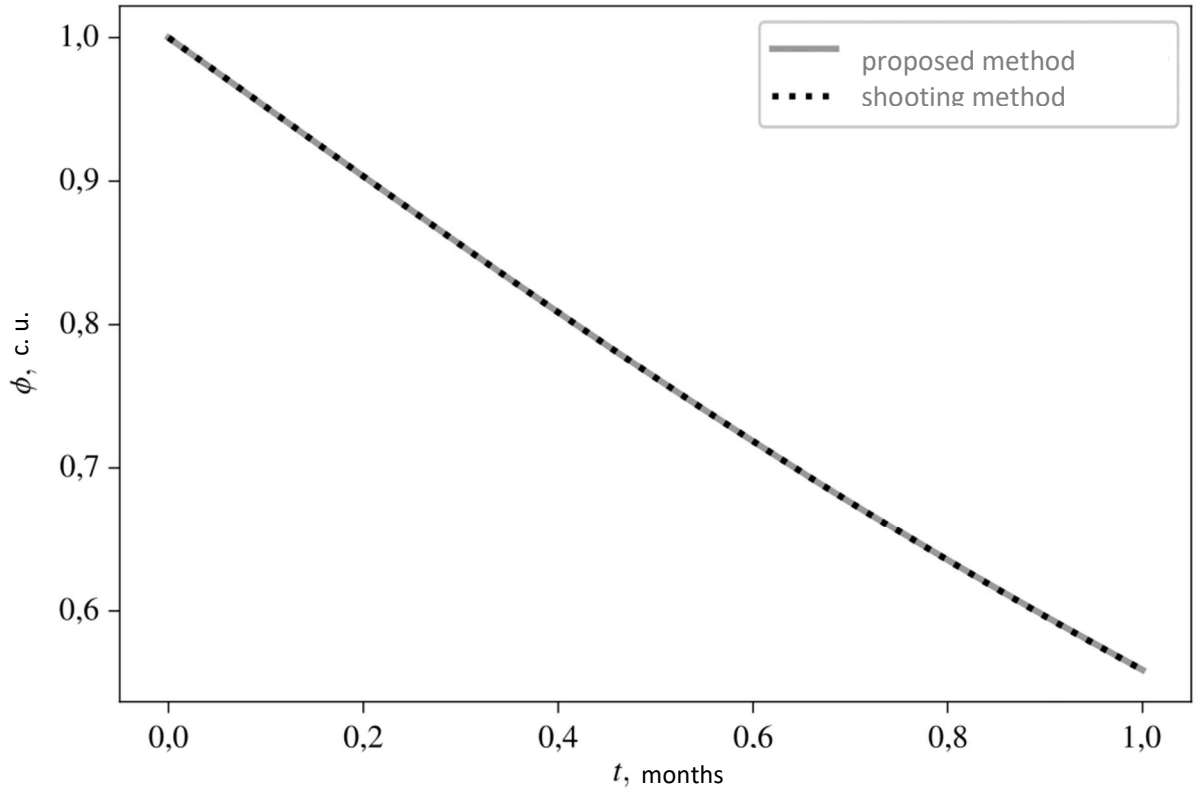
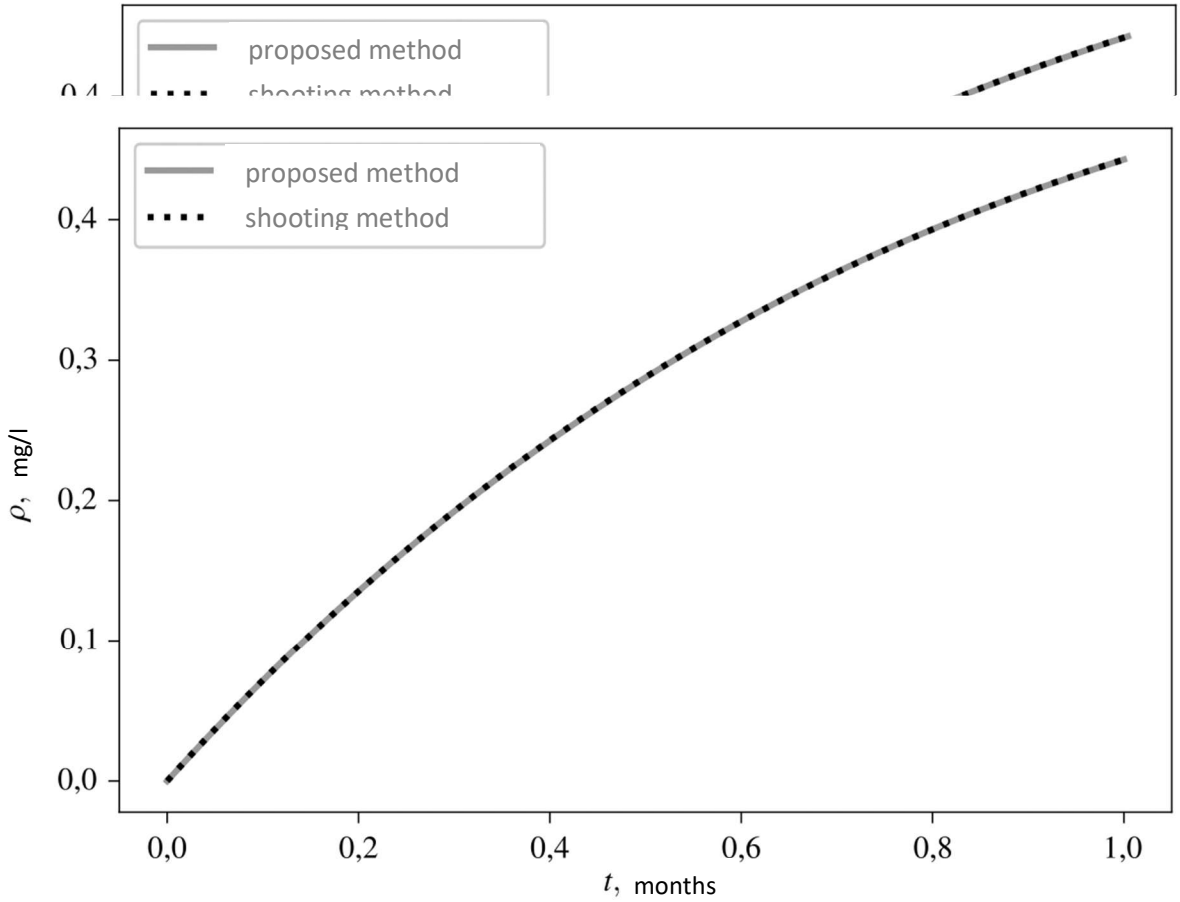


Fig. 2. Comparison of the dynamics of agent's funds for the problem of surface water quality control with a single agent (Example 2), calculated by the neural network method and the shooting method

This example illustrates the ability of the neural network method to solve the problem of sui
by



Fi
quality control with a single agent (Example 2), calculated by the neural network method and the shooting method

Example 3

This example considers the problem from Example 2, but on a longer time interval ($t_1 = 7$ months) and $z = 1.0$. If the modelled time interval is significant, then the solution cannot be obtained by the shooting method, the method diverges. For large time intervals, the shooting method converges slowly or diverges. For the shooting method, this problem is described in literature [Rao, 2009]. Therefore, in this example, the solution by the shooting method is not found. The solution is found using the neural network method.

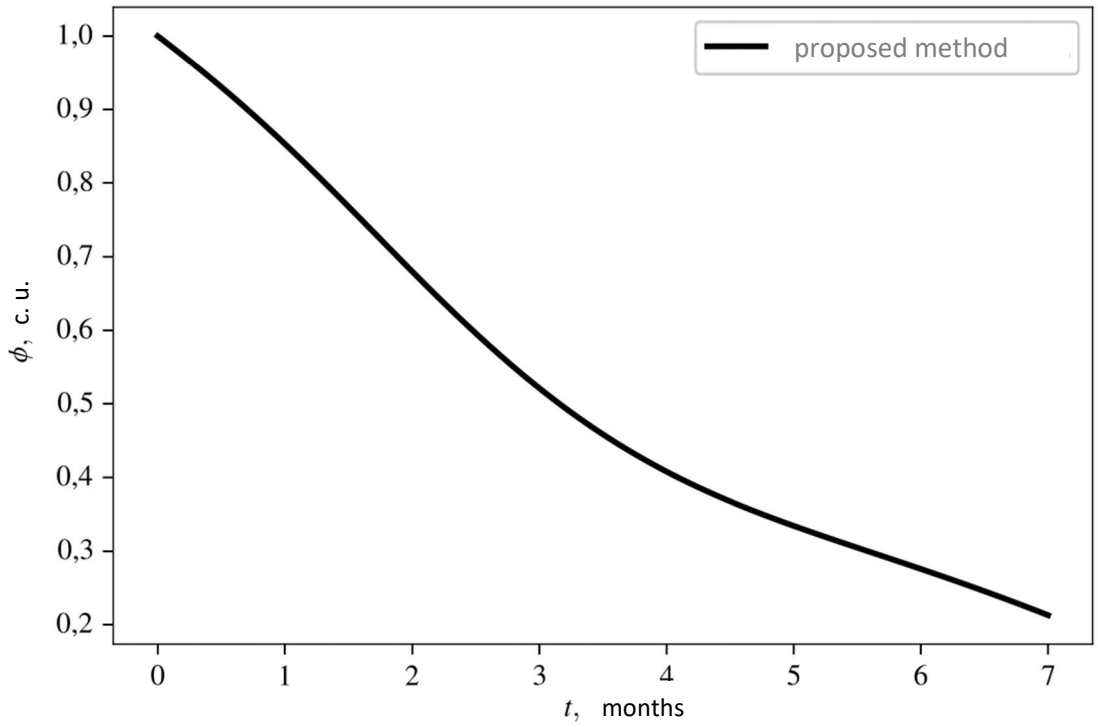


Fig. 4. Change in the value of agent's funds over time for the problem of surface water quality control with a single agent and increased time (Example 3). The calculations were carried out by the neural network method, other methods failed to obtain a solution.

A neural network is introduced, similar to the previous examples, with the number of neurons 4, 32, 128. The input of the network corresponds to the time, which is normalized as $t = \frac{t}{7} - 0.5$; thus, t lies in the interval $[-0.5;\ 0.5]$. The rest corresponds to Example 2. For training, a modification of the adam+Lookahead gradient descent method with cyclicLr strategy for choosing learning rate is used. As a result of training, the average value of L decreased to 0.001. Error values are obtained $\overline{\Delta} = 0.002;\ 0.002;\ 0.003;\ 0.002$ for $\phi(t), \rho(t), \lambda_1(t), \lambda_0(t)$, respectively. Figures 4 and 5 show the calculation results. This example demonstrates the possibility of studying the

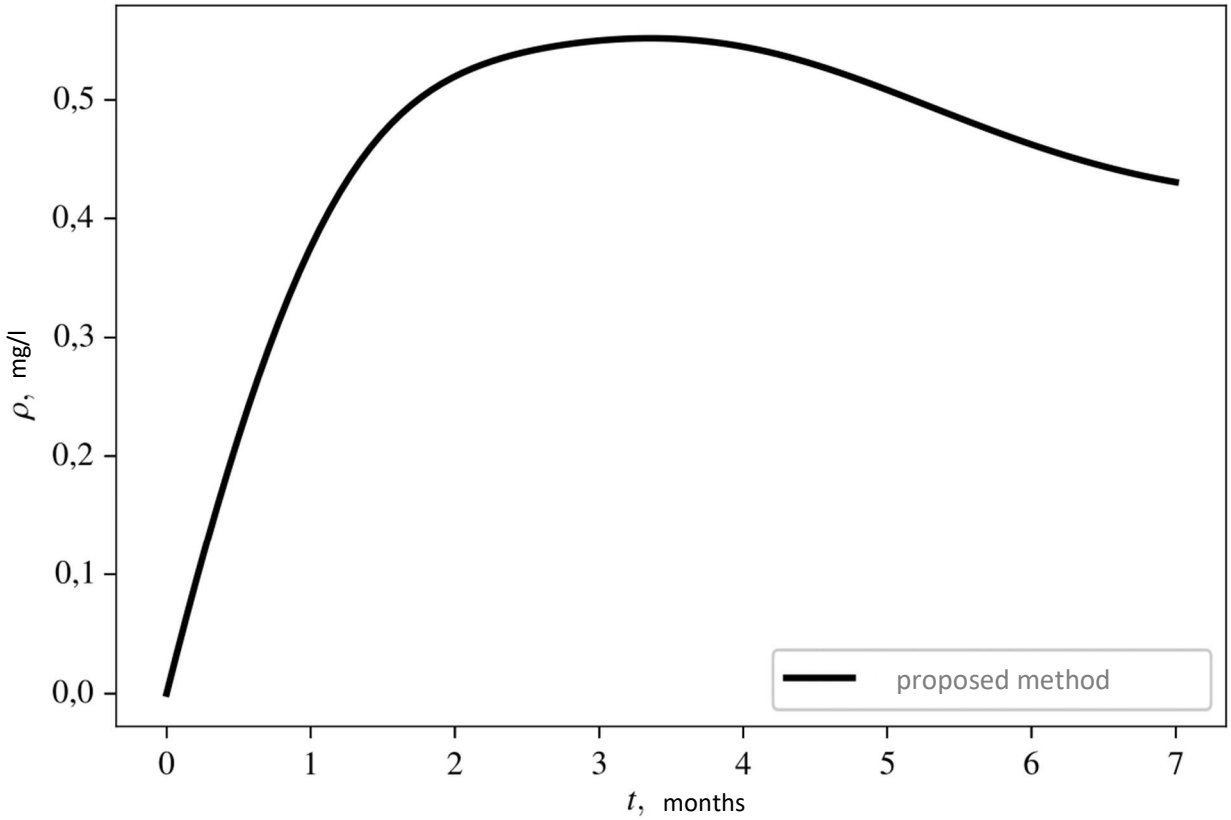


Fig. 5. Change in the concentration of pollutants over time for the problem of surface water quality control with a single agent and increased time (Example 3). The calculations were carried out by the neural network method, other methods failed to obtain a solution.

Example 4

In this example, problem (1)-(4) is studied for the case of two agents. By similar logic as in the previous examples, the expressions for optimal controls are obtained:

$$u_i{}^{opt} = \begin{cases} \text{argmax}\big(H_i(u_{min}), H_i(u_{max}), H_i(u_i{}^*)\big), & u^* \in [u_{min}, u_{max}], \\ \text{argmax}\big(H_i(u_{min}), H_i(u_{max})\big), & u_i{}^* \notin [u_{min}, u_{max}], \end{cases}$$

where

$$u_i{}^* = \frac{-\lambda_{i0}\rho_{max} + \rho v_i(1 - \eta_i + \eta_i\lambda_{ii})}{2\rho_{max}(1 - \eta_i + \eta_i\,\lambda_{ii})}$$

is the zero of the derivative of H_i with respect to u_i. Here the essential difference is that the values of H_i in the general case depend not only on the controls of the *i*-th agent, but also on the controls of other agents; thus, the problem of finding the Nash equilibrium for agents is obtained. The function of agents' optimal response is defined: $P(u)$: $\mathbb{R}^N \to \mathbb{R}^N, P_i(u) = u_i{}^{opt}$. Then the solution to the equation $P(u) = u$ will provide the required Nash equilibrium. In addition, since there are two agents, the system dynamics is described by three spatial variables ρ, ϕ_1, ϕ_2 and six introduced variables $\lambda_{10}, \lambda_{11}, \lambda_{12}, \lambda_{20}, \lambda_{21}, \lambda_{22}$. The obtained optimality conditions are used for solving the problem by the neural network method.

A neural network of three layers with sigmoid activation functions and linear activation on the output layer is introduced. It is denoted as *nn(t)*, and its outputs as $nn_{\phi_1}(t), nn_{\phi_2}(t), nn_{\rho}(t), nn_{\lambda_{10}}(t), nn_{\lambda_{11}}(t),$ $nn_{\lambda_{12}}(t), nn_{\lambda_{20}}(t), nn_{\lambda_{21}}(t), nn_{\lambda_{22}}(t), nn_{u_1}(t), nn_{u_2}(t)$, respectively. The input of the network corresponds to the time, which is normalized as $t = t - 0.5$; thus, *t* lies in the interval $[-0.5;\ 0.5]$. The number of neurons in the layers is 4, 32 and 128, respectively, counting from the input of the network. To satisfy the boundary conditions, a solution is sought in the following form: $\overline{nn}(t) = A(t) + F\big(t, nn(t)\big)$, where $A(t) = [\,\phi_{10}, \phi_{20}, \rho, 0, 0, 0, 0, 0, 0, 0, 0]$, $F(t) = \big[(t + 0.5)nn_{\phi_1}, (t + 0.5)nn_{\phi_2}, (t + 0.5)nn_{\rho}, (t - 0.5)nn_{\lambda_{10}}, \quad (t - 0.5)nn_{\lambda_{11}}, (t - 0.5)nn_{\lambda_{12}}, (t - 0.5)nn_{\lambda_{20}}, (t - 0.5)nn_{\lambda_{21}}, (t - 0.5)nn_{\lambda_{22}}, \frac{u_{max}}{(1+e^{-nn_{u_1}})}, \frac{u_{max}}{(1+e^{-nn_{u_2}})}\big]$.

A loss function that describes the deviation of the derivative of the neural network from the condition of the problem is introduced:

$$L = \left(\frac{\partial\overline{nn}}{\partial\phi_1}[\phi_1] - \dot{\phi}_1(\overline{nn}(t), u^{opt})\right)^2 + \left(\frac{\partial\overline{nn}}{\partial\phi_2}[\phi_2] - \dot{\phi}_2(\overline{nn}(t), u^{opt})\right)^2 + \left(\frac{\partial\overline{nn}}{\partial\rho}[\rho] - \dot{\rho}(\overline{nn}(t), u^{opt})\right)^2 +$$

$$+\left(\frac{\partial\overline{nn}}{\partial\lambda_{10}}[\lambda_{10}]-\dot{\lambda}_{10}(\overline{nn}(t),u^{opt})\right)^2$$
$$+\left(\frac{\partial\overline{nn}}{\partial\lambda_{11}}[\lambda_{11}]-\dot{\lambda}_{11}(\overline{nn}(t),u^{opt})\right)^2\left(\frac{\partial\overline{nn}}{\partial\lambda_{12}}[\lambda_{12}]-\dot{\lambda}_{12}(\overline{nn}(t),u^{opt})\right)^2+$$

$$+\left(\frac{\partial\overline{nn}}{\partial\lambda_{20}}[\lambda_{20}]-\dot{\lambda}_{20}(\overline{nn}(t),u^{opt})\right)^2$$
$$+\left(\frac{\partial\overline{nn}}{\partial\lambda_{21}}[\lambda_{21}]-\dot{\lambda}_{21}(\overline{nn}(t),u^{opt})\right)^2\left(\frac{\partial\overline{nn}}{\partial\lambda_{22}}[\lambda_{22}]-\dot{\lambda}_{22}(\overline{nn}(t),u^{opt})\right)^2+$$

$$+(P(nn(t)[u_1])-nn(t)[u_1])^2+(P(nn(t)[u_2])-nn(t)[u_2])^2.$$

As a training sample, a set of time points T chosen by a uniform partition of the interval [0, 1] is introduced. The example takes 5000 points, including the ends of the interval. Thus, the problem of minimizing the average value of L is solved:

$$\min_w \overline{L(t)}, \quad t \in T.$$

To solve it, a modification of the adam gradient descent method with cyclicLr strategy for choosing learning rate is used. The calculations were carried out with the following input data: $N = 2$, $t_0 = 0$,
$t_1 = 1$ month, $k = 0.5$, $z = [1.0;\ 1.0]$, $u_{max} = 0.9$, $u_{min} = 0$, $\rho_{max} = 1$ mg/l, $\phi_0 = [1.0;\ 1.5]$ c. u.,
$\rho_0 = 0$ mg/l, $R(\phi) = \phi$, $W(\phi) = \phi$, $C(u) = u^2$, the values of *v, η, β* varied. Table 1 shows the results of the calculations and the values of *v, η, β*. Thus, an increase in the fee for the discharge of pollutants can lead to a significant decrease in the agent's funds and their winnings. At the same time, a decrease in the fee for the discharge of pollutants leads to an increase in the concentration of pollutants in the water body. Therefore, in order to achieve sustainable development of the "agents – water body" system, it is necessary to introduce a regulatory body to set a fee for the discharge of pollutants. These conclusions correspond to the conclusions of another work [Reshitko, Ugolnitsky, Usov, 2020], where the calculations were carried out by the shooting method. In addition, over a time interval of 1 month, there is no excess of the maximum permissible concentration of pollutants.

Thus, Examples 1 and 2 illustrate the application of the method to problems with a known solution, with the solution by the neural network method corresponding to the exact solution in Example 1 and to the solution by the shooting method in Example 2. In addition, the error of the method calculated with respect to the exact solution and the error calculated by the proposed method are of the same order, which demonstrates the consistency of the proposed error estimation method. Examples 2, 3 and 4 provide calculations of the surface water quality control system for one and two agents of various types.

Table 1. Results of calculations performed by the neural network method for the problem of surface water quality control with two agents (Example 4)

J_1, mln RUB	J_2, mln RUB	β_1	β_2	η_1	η_2	v_1	v_2	$\phi_1(T)$, c. u.	$\phi_2(T)$, c. u.	$\rho(T)$, mg/l	max $\overline{\Delta}$	
0.3160	0.1053	0.5	0.5	0.5	0.5	1	1	0.847	0.9902	0.6154	0.0063	
0.3281	0.1072	0.5	0.5	0.5	0.5	0.9	0.9	0.8569	0.9915	0.6639	0.005	
0.3560	0.1114	0.5	0.5	0.5	0.5	0.7	0.7	0.8797	0.9953	0.6756	0.0054	
0.3882	0.1143	0.5	0.5	0.5	0.5	0.5	0.5	0.9063	0.9975	0.725	0.008	
0.4263	0.1235	0.5	0.5	0.5	0.5	0.3	0.3	0.938	1.005	0.7865	0.006	
0.4730	0.1246	0.5	0.5	0.5	0.5	0.1	0.1	0.9773	1.006	0.8522	0.0046	
0.0366	0.0113	0.5	0.5	0.95	0.95	0.9	0.9	1.1416	1.0739	0.708	0.0128	
0.1554	0.0441	0.5	0.5	0.8	0.8	0.7	0.7	1.0867	1.0448	0.7369	0.0194	
0.3210	0.0933	0.5	0.5	0.6	0.6	0.5	0.5	0.9793	1.0174	0.7518	0.0131	

0.5525	0.1663	0.5	0.5	0.3	0.3	0.3	0.3	0.7898	0.9647	0.7254	0.0118	
0.7119	0.2137	0.5	0.5	0.1	0.1	0.1	0.1	0.6678	0.9283	0.7132	0.0115	
0.0419	0.0146	0.0	0.0	0.95	0.95	0.9	0.9	1.7926	1.7779	0.848	0.0161	
0.1681	0.0546	0.25	0.25	0.8	0.8	0.7	0.7	1.3705	1.3597	0.8331	0.0066	
0.3154	0.0919	0.6	0.6	0.6	0.6	0.5	0.5	0.9194	0.9223	0.7556	0.0141	
0.5070	0.1590	0.8	0.8	0.3	0.3	0.3	0.3	0.6042	0.7123	0.6919	0.0146	

0.6223	0.1802	1	1	0.1	0.1	0.1	0.1	0.4486	0.5473	0.5924	0.0085

Conclusion

This paper presents an approach to solving problems of optimal control and differential games using neural networks. A description of the neural network approach is presented, including error estimation methods and neural network training methods. The method is based on existing methods for solving systems of differential equations and optimal control problems. At the same time, the paper proposes a method for quantifying the solution error, and one neural network is used to approximate all unknown functions.
The development of a neural network method for solving optimal control problems makes it possible to
expand the range of problems to be solved, since the method uses the optimal control theory in combination with numerical methods of neural networks.

Above are examples of using the proposed method. Examples 1 and 2 compare the solution found by the proposed method with the exact solution and the solution obtained by the shooting method.
The absolute error Δ and the error based on the solution defect $\overline{\Delta}$ have been calculated; since they are of the same order, $\overline{\Delta}$ can be used to estimate the error of the neural network approach. For Example 3, the exact solution is unknown, and the solution can't be obtained by the shooting method, however, the solution has been obtained using the neural network method with an acceptable error value $\overline{\Delta}$. Example 4 solves the differential game of two agents by the neural network method, and the optimal controls are found not analytically, but by training the neural network. A number of numerical experiments have been carried out, the results of which are consistent with those obtained earlier. Thus, it can be concluded that the method is applicable to solving various formulations of the problem of surface water quality control, and the following advantages can be distinguished:

1) in contrast to various modifications of the shooting method, the boundary conditions are satisfied accurately and directly by the architecture of the neural network before it is trained;

2) the resulting solution is continuous and differentiable;

3) the method allows calculations over larger time intervals compared to the shooting method;

4) the acceleration of calculations on video cards is possible;

5) it is possible to use one neural network to approximate the entire solution, which speeds up and simplifies calculations;

6) it is possible to quantify the error of the obtained solution using the solution defect.References

1. Adam: a method for stochastic optimization. URL: arxiv.org/abs/1412.6980
2. Andreeva E.A., Tsiruleva V.M. Mathematical modeling of optimal control of dynamic systems with neural networks. *Modelling, optimization and IT*, 2018, Vol. 6, No. 2, pp. 119-131.
3. Auzinger *W.* Error estimation via defect computation and reconstruction: some practical techniques. *JNAIAM*, 2011, Vol. 6, No. 1-2, pp. 15-27.
4. Cybenko G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 1989, No. 2, pp. 303-314.
5. Cyclical learning rates for training neural networks. URL: arxiv.org/abs/1506.01186
6. Google/jax. URL: github.com/google/jax
7. Gorbachenko V.I., Artyukhina E.V. Two approaches for solving PDE with radial basis neural networks. *Bulletin of Universities, Volga Region. Technical sciences*, 2007, No. 2, pp. 56-66.
8. Gym. URL: gym.openai.com
9. HIPS/autograd. URL: github.com/HIPS/autograd
10. Holsapple R., Venkataraman R. New, fast numerical method for solving two-point boundary value problems. *Journal of guidance control and dynamics*, 2004, No. 27, pp. 301-304.
11. Hua L. Solving a nonlinear two-point boundary value problem. *IEEE International Conference on Systems, Man, and Cybernetics*, 1992, pp. 1038-1042.
12. Kamien M. I., Schwartz N. L. Dynamic Optimization. The calculus of variations and optimal control in economics and management. Elsevier, 1991.
13. Kovalenko A.N., Chernomorets A.A., Petina M.A. Using neural networks to solve PDEs. *Scientific Bulletin of Belgorod State University. Ser. Economy. Computer science*, 2017, No. 9, pp. 103-110.
14. Lookahead optimizer: *k* steps forward, 1 step back. URL: arxiv.org/abs/1907.08610
15. Malkin V.A. Solving two-point BVP using non-gradient random search. *System analysis and applied informatics*, 2016, pp. 29-34.
16. Nazemi A., Karami R. A neural network approach for solving optimal control problems with inequality constraints and some applications. *Neural processing letters*, 2017, No. 45, pp. 995-1023.

17. NN-for-Optimal-Control. URL: github.com/ReshitkoM/NN-for-Optimal-Control
18. Pirogov S.P.; Ustinov N.N.; Smolin N.I. (2018), Mathematical Model of Stress-Strain State of Curved Tube of Non-Circular Cross-Section with Account of Technological Wall Thickness Variation. *IOP Conf. Ser.: Mater. Sci. Eng. 357*. URL: DOI 10.1088/1757-899X/357/1/012037
19. PyTorch. URL: pytorch.org
20. Rao A.V. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 2009, No. 135, pp. 497-528.
21. Reshitko M. A., Ougolnitsky G. A., Usov A. B. Numerical method for finding Nash and Shtakelberg equilibria in river water quality control models. *Computer Research and Modeling*, 2020, Vol. 12, No. 3, pp. 653-667.
22. SciPy. URL: www.scipy.org
23. Stetter H. J. The defect correction principle and discretization methods. *Numerische Matematik*, 1978, No. 29, pp. 425-443.
24. Sung N.H. A nonlinear shooting method for two-point boundary value problem. *Computers and mathematics with applications*, 2001, No. 42, pp. 1411-1420.
25. Surinskiy, D.O., Savchuk, I.V., Solomin, E.V., Kovalyov, A.A. (2019). PV-based energy-saving electro-optical converter development. *19th International Scientific Geoconference SGEM 2019. Conference proceedings*. Vol. 19, No. 4.1, pp. 427-434.
26. Sutskever I., Martens J., Dahl J., Hinton G. On the importance of initialization and momentum in deep learning. *Proceedings of the 30th international conference on machine learning*, 2013, pp. 1139-1147.
27. Yadav N., Yadav A., Kumar M. An introduction to neural network methods for differential equations. Springer Briefs in Applied Sciences and Technology, 2015.
28. Yize C., Yuanyuan S., Baisen Z. Optimal Control Via Neural Networks: A Convex Approach. ICLR Conference. 2019.
29. Zadunaisky P. E. On the estimation of errors propagated in the numerical integration of ordinary differential equations. *Numerische Matematik*, 1976, No. 27, pp. 21-39.
30. Zhou L., Hongming P., Feicheng W., Zhiqiang H., Liwei W. The expressive power of neural networks: a view from the width. NIPS. 2017.