This paper examines the use of neural network methods to solve inverse problems in the mechanics of elastic materials.

The aim is to design physics-informed neural networks that can predict the parameters of structural components, and the physical properties of materials based on a specified displacement distribution.

A key feature of the specified neural networks is the integration of differential equations and boundary conditions into the loss function calculation. This approach ensures that the error in approximating unknown functions has a direct impact on optimizing the network's weights. As a result, the resulting neural network approximations of unknown functions comply with the differential equations and boundary conditions.

To test the capabilities of the designed neural networks, inverse problems involving the bending of plates and beams have been solved, focusing on determining one or two unknown parameters. Comparison of predicted and exact values demonstrates high accuracy of the constructed neural network models, with a relative prediction error of less than 3 % across all cases.

Unlike analytical methods for solving inverse problems, the primary advantage of physics-informed neural networks is their flexibility when addressing both linear and nonlinear problems. For instance, the same network architecture can be employed to solve various boundary-value problems without modification. Compared to classical numerical methods, the parallelization capability of neural networks is inherently supported by modern software libraries.

Therefore, the application of physics-informed neural networks for solving inverse elasticity problems of plates and beams is effective, as evidenced by the achieved relative errors and the computational robustness of the method. In practice, the proposed solution can be used for relevant calculations during the design of structural elements. The developed software code can also be integrated into automated design systems or computer algebra systems

Keywords: physics-informed neural networks, inverse problems, geometric nonlinearity

UDC 004.8.032.26:517.954

DOI: 10.15587/1729-4061.2024.313795

DEVELOPING OF NEURAL NETWORK COMPUTING METHODS FOR SOLVING INVERSE ELASTICITY PROBLEMS

Anastasiia Kaliuzhniak
PhD
Department of Computer Science**

Oleksii Kudin
Corresponding author
PhD, Associate Professor
Department of Software Engineering**
E-mail: avk256@gmail.com

Yuriy Belokon
Doctorof Technical Sciences, Professor*

Dmytro Kruglyak
PhD, Associate Professor*

*Department of Metallurgical Technologies, Ecology and Technological Safety**
**Zaporizhzhia National University
Universytetska str., 66,
Zaporizhzhia, Ukraine, 69600

Received 09.10.2024
Received in revised form 02.12.2024
Accepted 12.12.2024
Published 20.12.2024

How to Cite: *Kaliuzhniak, A., Kudin, O., Belokon, Y., Kruglyak, D. (2024). Developing of neural network computing methods for solving inverse elasticity problems. Eastern-European Journal of Enterprise Technologies, 6 (7 (132)), 45–52. https://doi.org/10.15587/1729-4061.2024.313795*

1. Introduction

Modern materials, in particular, intermetallics, functional-gradient and composite materials are common in such fields as the aviation and aerospace industry, automotive industry, energy, etc. Effective design of structural elements from such heterogeneous materials is a difficult task and requires the use of modern methods for solving elasticity problems. In general, the use of computer simulations and virtual computing experiments makes it possible to reduce the costs of field studies, which may be associated with the destruction or damage of technical objects.

Engineering problems arising from such computer simulation are often reduced to the analysis of mathematical models, which are described by linear and nonlinear differential equations and the corresponding conditions on the boundaries of the areas of definition of technical objects [1]. Boundary value problems play a key role in the theory of elasticity and construction mechanics, taking into account the physical characteristics of structural materials.

Approximate methods by Ritz, Galerkin, of collocations and finite elements are usually used to solve boundary value problems [2]. The directions of "scientific machine learning" (SciML) or "Physics-informed machine learning" (PIML) are also becoming widespread [3, 4]. In these approaches, neural networks are used to approximate unknown functions [5]. Neural network variants of the Ritz, Galerkin, collocation and other similar methods were developed [6, 7]. As a result of the operation of these algorithms, a neural network with physical information (Physics-Informed Neural Networks, PINN) is formed, which satisfies differential equations and boundary conditions [8].

Solving inverse problems for differential equations is of great practical importance. In various fields of science and engineering, these problems make it possible to obtain important information about systems described by differential equations. For example, in mechanics problems, it is possible to determine the parameters of structures that satisfy a certain measured or projected value of the stress-strain state.

Analytical and numerical methods are standard approaches for finding solutions to inverse problems. Analytical methods usually require complex mathematical transformations and do not always make it possible to obtain a solution in an explicit form. Numerical methods, on the other hand, can require significant computational resources to reach convergence. The use of neural network methods can reduce the marginal problem to the problem of optimizing the weight coefficients of neural networks. In the general case, such an optimization problem is effectively solved by methods of gradient descent and error backpropagation when training neural networks.

Scientific research on the development of computational methods for solving differential equations and their systems is important for the theory of neural networks and for practical engineering applications. An actual direction is the expansion of the use of neural networks for solving non-linear physical and engineering problems.

2. Literature review and problem statement

Paper [9] discusses the technique for combining the finite element method with neural networks for solving direct and inverse boundary value problems. As a result, the work formulates a hybrid model for solving direct and inverse problems. The finite element method and physics-informed neural networks are used to form a joint loss function when solving direct problems. Losses are optimized during system training using the method of error backpropagation. In the inverse statement, the problem of identifying the flow rate-dependent coefficients of liquid bearings is solved. The disadvantage of the work is the lack of learning curves of the hybrid systems and hyperparameters of the neural networks that were used. Probably, this is due to the fact that the authors mostly paid attention to the theoretical justification of the method. The lack of data on the structure and hyperparameters of the neural networks used in the work does not make it possible to reproduce the results reported in the paper. An option to overcome these difficulties is the use of standardized libraries, such as DeepXDE [10].

The Python library for solving differential equations DeepXDE, which is an implementation of the approach based on physics-informed neural networks, is considered in [10]. A feature of the library is the use of the method of adaptive refinement of the solution based on residuals. The finite element method is used for comparison with the results obtained in the work. Neural networks were used to solve ordinary differential equations and equations in partial derivatives, the inverse problem of differential equations in partial derivatives, as well as to approximate a given function. The disadvantage of the work is insufficient coverage of the software architecture of the library. The reason for this may be that the library is publicly available and has a rather complex structure, and therefore it was inappropriate to describe the software architecture in the paper.

The method of solving the Hirota equation with variable coefficients is considered in [11]. A data-driven solution of direct and inverse problems is constructed. Local adaptive activation functions were used to design one of the variants of physics-informed neural networks. The effectiveness of the improved PINN algorithm is demonstrated for parameter prediction under different noise intensities. Different strategies of parameter regularization and corresponding weighting factors were used. The results obtained in the paper confirm that forward and inverse problems, including data-driven variable coefficient function discovery, can be solved by deep learning. The disadvantage of the work is the lack of an open software implementation of the described approaches. This can be explained by the fact that software code for academic research usually needs refactoring for further use by a wide range of specialists.

Direct and inverse problems of Lorentz and Ressler chaotic and hyperchaotic systems are considered in [12]. Neural Fourier operators are used to solve direct problems. The sparse identification of nonlinear dynamics (SINDy) algorithm is used for solving inverse problems. The influence of the depth of the Fourier layers and various activation functions on the accuracy of the networks was analyzed. The disadvantage of the work is the lack of analysis of the possibility of using other types of neural operators, for example, DeepONet or convolutional neural operator. This is probably due to the use of the Fourier transform in the operator layers, which make it possible reduce the volume of data for modeling nonlinear systems.

The inverse problems of groundwater flow modeled using the Richards partial differential equation are considered in [13]. The PINN of the network is used for solving. Inverse problems are solved by reformulating the loss function of a deep neural network. Such a loss function must simultaneously aim to satisfy the measured values and the unknown values in a set of points distributed in the problem domain. The inverse problem data set was generated using a finite difference method applied to solve the forward problem in the given domain. The shortcoming of the work is the lack of analysis of the impact of various methods of data generation for the inverse problem on the accuracy of determining process parameters. This can be explained by the fact that such computational experiments require significant resources. At the same time, the increase in accuracy when choosing a certain method for data generation may be insignificant for this class of problems.

In [14], the neural network architecture of direct signal propagation was used to model blood flow in arteries. For this purpose, the Navier-Stokes equation and the Windkessel model of hemodynamics were used. The PINN network was used to estimate the time-dependent velocity and pressure of blood flow in the aorta. Average values of velocity and pressure, which were measured using phase-contrast magnetic resonance imaging, were used as input data. The work contains a link to the open source code of the publication. The shortcoming of the paper is the insufficient description of the set of magnetic resonance imaging images and the lack of experiments with real clinical data. This can be explained by the complexity of preparing and processing medical images.

Work [15] reports the use of physics-informed neural networks to approximate the Euler equations that model high-speed aerodynamic flows. The authors solved direct and inverse problems in one-dimensional and two-dimensional domains. For the direct problem, the Euler equation and initial/boundary conditions are used to formulate the loss function. One-dimensional Euler equations with smooth solutions and with solutions that have a contact discontinuity are solved. It is shown that a stable solution can be obtained using distributed points randomly clustered around discontinuities. The shortcoming of the work is the insufficient analysis of adaptive methods for determining the points of the region in which the boundary conditions and the differential equation are satisfied. Such an analysis would make it possible to understand the limits of application of the proposed approach. This can be explained by the fact that reasonable accuracy was obtained using a random distribution of points, and further analysis was considered inappropriate by the authors.

So, from our review of the literature, it can be concluded that most works consider the solution to inverse problems of the dynamics of liquid and gas flows. At the same time, the application of the above neural network methods to solving linear and nonlinear elasticity problems is not covered in detail. In particular, effective methods for modeling the inverse problems of plates and beams would make it possible to determine the parameters of structural elements based on known displacements. Therefore, it is advisable to conduct a study aimed at applying physics-informed neural networks to solve linear and nonlinear inverse problems of the elasticity of plates and beams.

3. The aim and objectives of the study

Our goal is to design physics-informed neural networks for predicting parameters of structural elements and physical characteristics of materials based on a predefined displacement distribution. This will make it possible to introduce appropriate neural network methods into modern automated design systems for rapid assessment of structural characteristics.

To achieve the goal, the following tasks have been formulated:

– to develop a software implementation of the physics-informed neural network method;

– to apply the method of neural networks to solve inverse linear and nonlinear problems of the mechanics of elastic bodies.

4. The study materials and methods

The object of our research is neural network methods for solving inverse problems of the mechanics of elastic bodies.

The hypothesis of the study assumes that the use of physics-informed neural networks is effective in solving inverse problems of elasticity not only in a linear but also in a nonlinear statement.

This paper considers the bending problems of round plates and beams with a circular cross-section. In this case, simplified hypotheses of the linear theory of elasticity, as well as the geometrically nonlinear theory, are accepted. This makes it possible to simplify the general statement of the problem of the theory of elasticity. However, due to the adopted simplified hypotheses, it is possible to compare our results with published works.

In the PINN method, the neural network is trained by integrating physical laws described by differential equations directly into the learning process [7]. The result of PINN work can be represented as a complex composition of functions where, in addition to classic neural operations, the network takes into account differential equations and boundary conditions. As a result, the neural network approximates the unknown function and satisfies the differential equation and boundary conditions.

The general PINN equation for the case of inverse problems takes the form:

$$\hat{u}(x,\theta)=f_L\Big(f_{L-1}\big(\ldots f_2\big(f_1(x,\theta)\big)\big)\Big), \qquad (1)$$

where x is an input vector of data, in the general case is a spatial variable (depending on the dimensionality), $\hat{u}(x,\theta)$ is an approximate solution to a differential equation modeled by a neural network, f_l is a function of the l-th layer, which is a combination of linear transformation and activation, θ is a set of parameters characterizing the system, for example, coefficients of a differential equation or boundary conditions.

Unknown parameters of the system in problems of elasticity are usually used to determine the physical constants of the material (Young's modulus, Poisson's ratio) or the geometric dimensions of structural elements.

Each function f_l can be described as a linear transformation involving a weighted sum of the inputs with the addition of a shift, as well as the application of a nonlinear activation function σ:

$$f_l(x,\theta)=\sigma\big(W_l\cdot f_{l-1}(x,\theta)+b_l\big), \qquad (2)$$

where W_l is the matrix of neural network weights for the lth layer, b_l is the shift vector, and σ is the activation function.

The generalized loss function $\mathrm{E}(W,b,\theta)$ of the PINN network, which depends on the weighting coefficients of the neural networks, the shift, and the unknown parameters:

$$\begin{aligned}&\mathrm{E}(W,b,\theta)=\mathrm{E}_{pde}(W,b,\theta)+\\&+\mathrm{E}_{boundary}(W,b,\theta)+\mathrm{E}_{initial}(W,b,\theta),\end{aligned} \qquad (3)$$

where $\mathrm{E}_{pde}(W,b,\theta)$ is the error of the differential equation, $\mathrm{E}_{boundary}(W,b,\theta)$ is the error of the boundary conditions, $\mathrm{E}_{initial}(W,b,\theta)$ is the error of the initial conditions.

PINN uses the automatic differentiation method to calculate the derivatives of the neural network not only with respect to the unknown variables but also with respect to the parameters θ, which allows the parameters to be updated in the learning process using gradient methods.

Therefore, the network learns both to solve the differential equation and to find the optimal values of the parameters θ that minimize the loss function, i.e., the neural network solves the inverse problem during training.

For all inverse problems considered below, the solutions to direct problems of bending plates and beams from works [8, 16] are input data. In this case, the prediction of the neural network is compared with the corresponding distributions of movements. The resulting error is, in fact, the error of $\mathrm{E}_{boundary}(W,b,\theta)+\mathrm{E}_{initial}(W,b,\theta)$. Next, the error of the differential equation is calculated. During the training of the network, the parameters θ, which were entered during the initialization of the neural network as its weighting factors, are also adjusted.

To form a training and testing sample in a given area, several strategies for choosing collocation points are used in practice, in which the error of compliance with the differential equation is calculated [17]. In particular, methods of non-adaptive uniform and adaptive non-uniform selection of points are considered in [17]. The advantages of the adaptive method are the possibility of condensing the number of points near areas with features (holes, stress concentrators, etc.). At the same time, uniform sampling is fast and provides satisfactory accuracy for homogeneous structural elements. In this work, uniform sampling with different steps is used for training and test data.

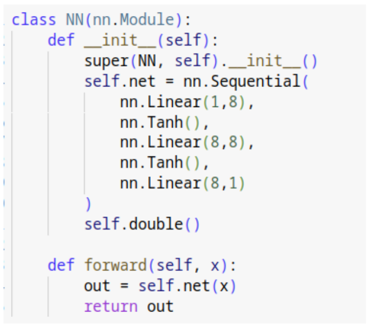
5. Results of implementing the physics-informed neural networks for solving inverse problems of the mechanics of elastic bodies

5. 1. Results of the software implementation of the method of physics-informed neural networks

The software implementation of the neural network method consisted in creating appropriate data structures and functions for determining neural networks and the conditions of the problems being solved. Two classes are implemented: NN and Net_inv using PyTorch and Numpy libraries. The NN class defines the neural network, the number of layers and neurons. Fig. 1 shows the software code for defining a neural network with three fully connected layers of eight neurons each. The hyperbolic tangent activation function ensures the nonlinearity of the neural network model, which usually leads to a better approximation of dependences in the data.

The Net_inv class is implemented to realize the PINN method. This class defines the distribution of data points on which the model is trained and the method for calculating the loss function of the neural network. Fig. 2 shows a code fragment that defines the network parameters that are adjusted during gradient descent training. In particular, the parameter lambda_value is included in the internal weights of the neural network. Therefore, the lambda_value will be optimized during network training to match the differential equation and the input displacement distribution from the forward problem solution. The uf parameter of the class constructor contains the value of the unknown function at the specified points, i.e., it is the solution to the direct problem at the specified points of the domain of definition [8].

The method of the Net_inv class, which calculates the loss function of the neural network, is shown in Fig. 3. The peculiarity of PINN is the loss function, which consists of the approximation error at the points of the domain of definition and the error of the differential equation.

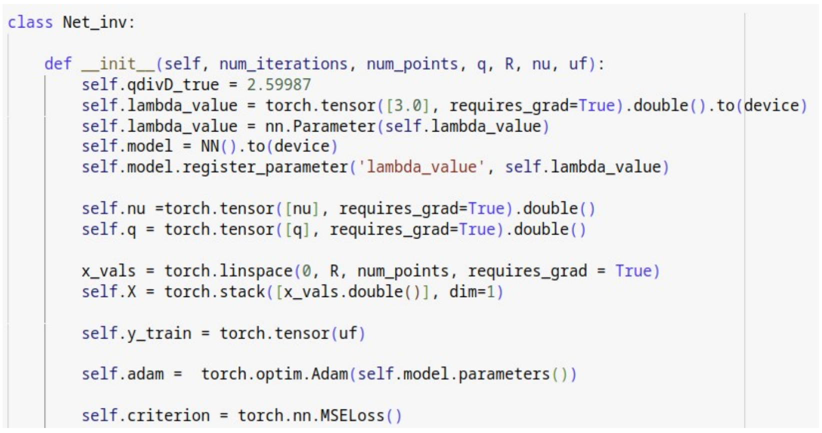


```
class NN(nn.Module):
    def __init__(self):
        super(NN, self).__init__()
        self.net = nn.Sequential(
            nn.Linear(1,8),
            nn.Tanh(),
            nn.Linear(8,8),
            nn.Tanh(),
            nn.Linear(8,1)
        )
        self.double()

    def forward(self, x):
        out = self.net(x)
        return out
```

Fig. 1. The class that defines the neural network of the PINN method

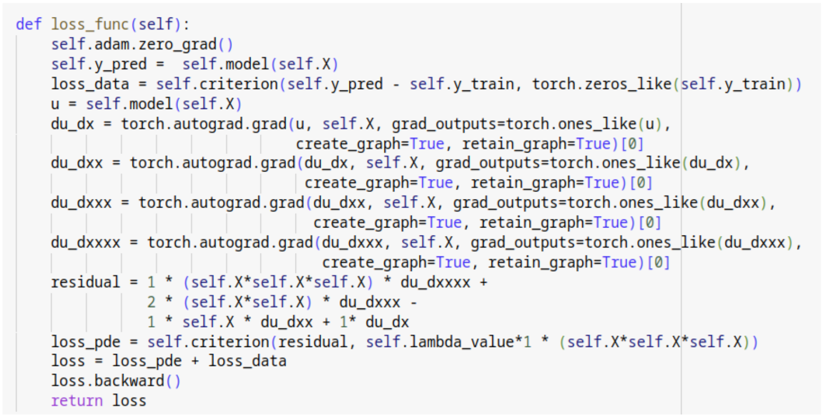
The value of the unknown function is calculated as the result of the prediction of the neural network on a given set of points. After that, numerical differentiation, and substitution of the obtained approximate values in the differential equation takes place. Also, the error of the network at the points of the definition domain is additionally calculated relative to the values of the solution to the direct problem. The resulting deviations are added to the general error of the neural network and the method of error backpropagation is used to optimize the weighting coefficients.

The software implementation of the proposed methods and the results of solving some problems are given in [18].



```
class Net_inv:

    def __init__(self, num_iterations, num_points, q, R, nu, uf):
        self.qdivD_true = 2.59987
        self.lambda_value = torch.tensor([3.0], requires_grad=True).double().to(device)
        self.lambda_value = nn.Parameter(self.lambda_value)
        self.model = NN().to(device)
        self.model.register_parameter('lambda_value', self.lambda_value)

        self.nu =torch.tensor([nu], requires_grad=True).double()
        self.q = torch.tensor([q], requires_grad=True).double()

        x_vals = torch.linspace(0, R, num_points, requires_grad = True)
        self.X = torch.stack([x_vals.double()], dim=1)

        self.y_train = torch.tensor(uf)

        self.adam =  torch.optim.Adam(self.model.parameters())

        self.criterion = torch.nn.MSELoss()
```

Fig. 2. The Net_inv class containing the implementation of the PINN method



```
def loss_func(self):
    self.adam.zero_grad()
    self.y_pred =  self.model(self.X)
    loss_data = self.criterion(self.y_pred - self.y_train, torch.zeros_like(self.y_train))
    u = self.model(self.X)
    du_dx = torch.autograd.grad(u, self.X, grad_outputs=torch.ones_like(u),
                                create_graph=True, retain_graph=True)[0]
    du_dxx = torch.autograd.grad(du_dx, self.X, grad_outputs=torch.ones_like(du_dx),
                                 create_graph=True, retain_graph=True)[0]
    du_dxxx = torch.autograd.grad(du_dxx, self.X, grad_outputs=torch.ones_like(du_dxx),
                                  create_graph=True, retain_graph=True)[0]
    du_dxxxx = torch.autograd.grad(du_dxxx, self.X, grad_outputs=torch.ones_like(du_dxxx),
                                   create_graph=True, retain_graph=True)[0]
    residual = 1 * (self.X*self.X*self.X) * du_dxxxx +
               2 * (self.X*self.X) * du_dxxx -
               1 * self.X * du_dxx + 1* du_dx
    loss_pde = self.criterion(residual, self.lambda_value*1 * (self.X*self.X*self.X))
    loss = loss_pde + loss_data
    loss.backward()
    return loss
```

Fig. 3. Loss function of the PINN method

5. 2. Results of using physics-informed neural networks for solving inverse linear and nonlinear problems of the mechanics of elastic bodies

The developed software implementation of physics-informed neural networks is applied to solving linear and nonlinear problems of bending plates and beams. These problems are test and intended for verification of the PINN method on well-known models. Linear statements assume the presence of exact solutions, and nonlinear ones – approximate solutions published in the literature. For the considered problems, the results of solving direct statements using physics-informed neural networks are given in [8]. Approximate solutions of direct statements are used as input data to determine unknown parameters when solving inverse problems.

Problem 1. Bending of a circular plate with a pinched contour under the action of a distributed transverse constant load. The differential equation of deflection $w(r)$ [8]:

$$\frac{d^4w}{dr^4}+\frac{2}{r}\frac{d^3w}{dr^3}-\frac{1}{r^2}\frac{d^2w}{dr^2}+\frac{1}{r^3}\frac{dw}{dr}=\frac{q}{D}, \tag{4}$$

where q – value of the distributed transverse load; D – cylindrical stiffness of the plate; r is a spatial coordinate in the polar coordinate system.

Boundary conditions were used in the following form [8]:

$$\frac{dw}{dr}(a)=0,\ \ \frac{dw}{dr}(0)=0,\ \ w(a)=0, \tag{5}$$

where a is the radius of the circular plate.

Parameters of transverse bending of a round single-layer plate: plate thickness h=18·10^{-3} m, radius a=0.4 m, shear modulus and Poisson's ratio of the material – G=2.77·10^4 MPa and ν=0.3, respectively, distributed load q=0.5 MPa.

The unknown parameter was the value of q/D, that is, the inverse problem was to find such a ratio of distributed load to cylindrical stiffness that corresponded to the input deflection distribution.

The convergence of the values of the unknown parameter is shown in Fig. 4. The change in the value of q/D during training of the neural network model is shown.

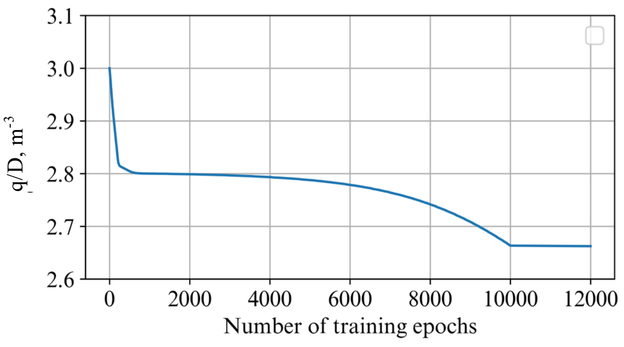


Fig. 4. Convergence of the solution to inverse problem 1

Problem 2. Bending of a circular plate with a hinged contour under the action of a distributed transverse permanent load. The differential equation coincides with the equation of problem 1, and the boundary conditions are:

$$w(a)=0,\ \ -D\left(\frac{d^2w}{dr^2}+\frac{\nu}{r}\frac{dw}{dr}\right)=0. \tag{6}$$

The convergence of forecasting the unknown parameter q/D when solving the inverse problem is shown in Fig. 5.

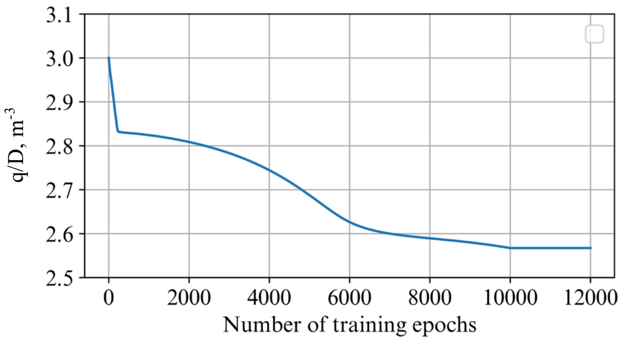


Fig. 5. Convergence of the solution to inverse problem 2

The unknown parameter, as in problem 1, was the ratio q/D. Therefore, the optimal values of the load and characteristics of the material of the round plate were determined, satisfying the input deflection distribution.

Problem 3. Bending of a beam of circular cross section with fixed ends under the action of a distributed transverse load. The differential equation of bending took the form [8]:

$$\frac{d^4w}{dx^4}=\frac{q}{EI}, \tag{7}$$

where q – value of the distributed transverse load; E – Young's modulus; I – moment of inertia of the beam; x – spatial coordinate in the Cartesian coordinate system.

Boundary conditions:

$$w(L)=0,\ \ w(0)=0,\ \ \frac{dw}{dx}(0)=0,\ \ \frac{dw}{dx}(L)=0, \tag{8}$$

where L is the length of the beam.

When solving the direct problem [8], the following geometric and physical parameters were adopted: length L=2 m, concentrated load force q=1 N, Young's modulus E=72 GPa, diameter of the circular section d=0.005 m.

In this case, the inverse problem of determining the Young's modulus E of the beam was solved. The change in the approximate values of the unknown parameter E with increasing training epochs is shown in Fig. 6.

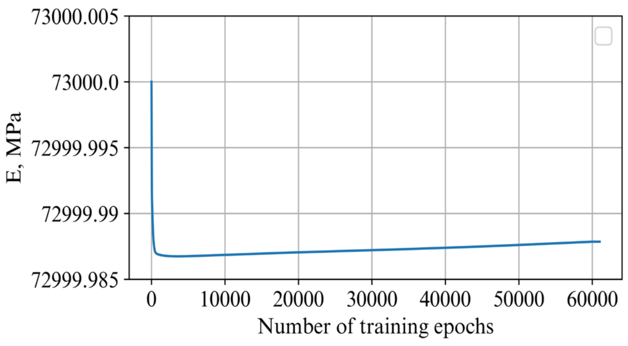


Fig. 6. Convergence of the solution to inverse problem 3

Problem 4. Bending of a beam with one fixed end and one free end [8].

The differential equation was used similarly to problem 3. The boundary conditions were as follows:

$$w(0)=0,\ \ \frac{dw}{dx}(0)=0. \tag{9}$$

In this case, the parameter determined when solving the inverse problem was the ratio q/EI. The convergence of the values of the unknown parameter is shown in Fig. 7

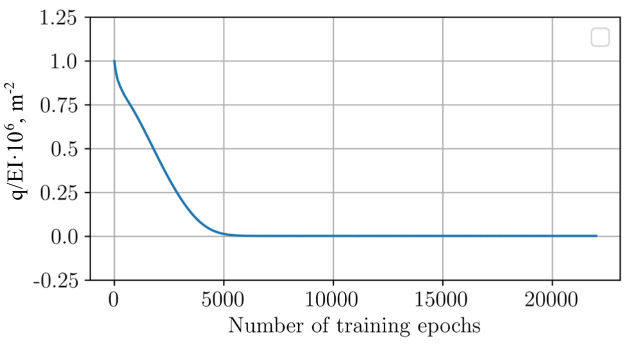


Fig. 7. Convergence of the solution to inverse problem 4

Problem 5. Bending of a beam with one fixed end in a geometrically nonlinear setting [8, 16]. The nonlinear differential equation in this case took the form:

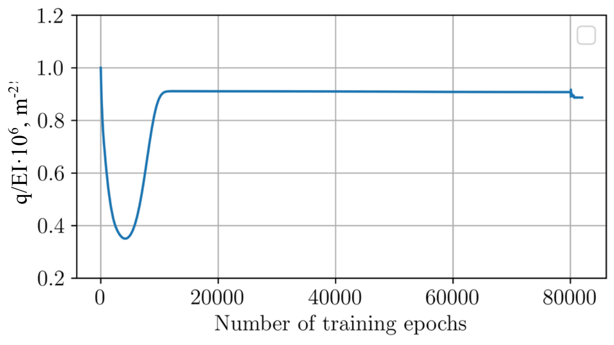
$$\frac{d^2w}{dx^2}=\frac{M(x)}{EI}\left(1+\left(\frac{dw}{dx}\right)^2\right)^{3/2}. \tag{10}$$

When solving the direct problem of beam bending with a concentrated load at the free end, the following parameters were used: length L=1 m, concentrated load force q=30 N, Young's modulus E=70 GPa, diameter of the circular section d=0.01 m [8].

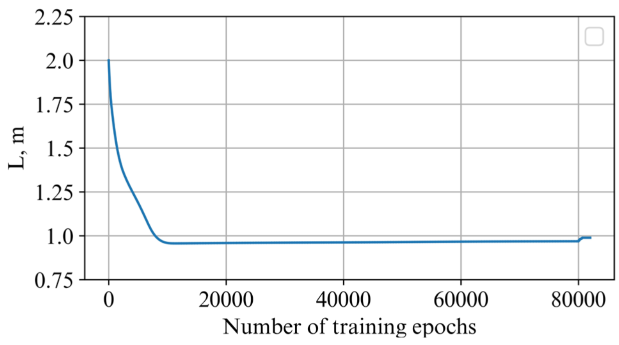
In this case, the inverse problem of determining two parameters of the beam – the q/EI ratio and the length L – was solved.

The change in the values of the unknown parameters of the ratio q/EI and the length L with an increase in the training epochs of the network is shown in Fig. 8.

Comparison of the values of parameters used when solving direct problems [8] and the values obtained when solving inverse statements is given in Table 1.



a



b

Fig. 8. Convergence of the solution to inverse problem 5: *a* – parameter q/EI; *b* – parameter L

Table 1

Comparison of exact and approximate parameter values

Parameter\Problem No.	1	2	3	4	5	
Unknown parameters	q/D	q/D	E	q/EI	q/EI	L
Exact value	2.600	2.600	72,020.0	0.00267	0.873	1.0
Approximate value	2.662	2.567	72,999.988	0.00266	0.887	0.989
Relative error, %	2.385	1.269	1.361	0.374	1.604	1.100

When performing computational experiments, neural networks of direct signal propagation with several internal layers (from 3 to 8) were used. The number of neurons in each layer varied from 8 to 16. Adam and LBFGS optimizers were used. The loss function is the root mean square error. Neural networks of this structure can learn effectively even in systems with only a central processor. The use of graphics processors significantly reduces the network training time. The models built were tested on a uniform grid of collocation points, but with a different step than during training.

6. Discussion of results based on applying the neural network methods for solving inverse elasticity problems

Our results of determining the parameters of structural elements when solving inverse problems testify to the satisfactory convergence of predictions of physics-informed neural networks to the exact values of the parameters.

The relative error (Table 1) of all test problems is less than 3 %, that is, the proposed approach is accurate enough for engineering calculations. One should note a relatively high error for problem 1 (2.385 %) in the case of bending a round plate with a pinched contour. This may indicate the need to select more layers of the neural network to provide a better approximation of the unknown function or to tune the hyperparameters of the network. Satisfactory results of the application of this method for solving the inverse problem in the case of geometrically nonlinear bending of the beam (less than 2 %) were obtained. This may indicate the prospects of using this approach for structural elements modeled using other nonlinear theories of elasticity.

In the considered inverse problems, the parameters characterizing the physical or geometric properties of the systems are unknown. Fig. 4–8 demonstrate that the values stabilize at a certain level already after ten thousand epochs of learning the neural network and almost do not change further. Therefore, the proposed approach is computationally robust. This can be explained by the effectiveness of methods for optimizing the weight coefficients of neural networks since the original inverse problem reduces to an optimization problem.

In contrast to [9, 11], our work uses a relatively simpler structure of physics-informed neural networks, which made it possible to obtain solutions of satisfactory accuracy for the considered classes of problems without the need to use additional resources for calculations.

The proposed software implementation of the method of physics-informed neural networks differs from [10] in that it has greater flexibility and the possibility of adaptation for solving specific boundary value problems. This is achieved by the fact that the classes implemented in the work use the PyTorch library directly. At the same time, the DeepXDE library offers its own interface for solving differential equations, which can be limited in settings.

In [12], neural operators were used, which make it possible to obtain a generalized solution to problems for arbitrary values of system parameters. However, information about physical relationships is not used to train neural operators. This can lead to solutions that do not satisfy the physical meaning of the problem. The difference of this work is that the physical equations are part of the loss function of neural networks, which leads to a correct solution on a small data set. Thus, the training sample for the considered problems consisted of 200–500 data points.

In contrast to [13–15], our work solves problems related to the statics of plates and beams. This type of problem is important when designing structural elements. The resulting solution demonstrates how the structure will behave under constant loads, ensures optimal use of materials, and evaluates possible deformations.

The method of physics-informed neural networks has certain advantages compared to analytical and numerical approaches.

For many real-world problems, especially for nonlinear or multidimensional systems, an analytical solution can be extremely difficult or impossible. Therefore, analytical methods often work only for special classes of equations (linear or simple nonlinear), and they are less effective for problems with complex or nonlinear interactions, especially in problems with heterogeneous materials or geometries [19].

On the other hand, classical numerical methods, such as the finite element method or the finite difference method, can solve very complex problems, including nonlinear, inhomogeneous, multidimensional systems, where analytical methods do not work. However, these algorithms are usually quite complex for software implementation and depend on the accuracy of discrete grids [20].

PINN combines neural networks and physical laws by integrating differential equations and boundary conditions directly into the learning process of neural networks. Unlike [19], in which analytical approaches to solving inverse problems are considered, neural network methods have no theoretical limitations on the type and complexity of boundary value problems. This is a new approach to solving both direct and inverse problems, the main advantage of which is its flexibility in solving linear and nonlinear problems. So, in particular, the same network can be used to solve different boundary value problems without changing the network architecture. And the use of modern hardware capabilities, for example, graphics and tensor processors, make it possible to significantly increase the speed of algorithms. Unlike classical numerical methods [19], the possibility of parallelizing neural networks is already included in modern software libraries.

Therefore, the use of physics-informed neural networks for solving inverse problems related to the elasticity of plates and beams is effective in view of the obtained values of relative errors and the computational stability of the method.

Our results complement the application of the method for solving inverse problems described in the literature, in particular, in terms of determining the parameters for nonlinear systems. The introduction of physical dependences and unknown parameters into the network structure make it possible reduce the inverse problem to an optimization problem, which is effectively solved by gradient descent methods. The software code published in the open repository can be freely implemented as modules of automated design systems or computer algebra.

Currently, the limitations of this work are the ability to solve only static problems in the theory of elasticity.

The disadvantage of the study is the lack of automated setting of hyperparameters of neural networks.

Further development of this method may involve its implementation for solving problems of dynamics and stability with automatic adjustment of hyperparameters of neural networks.

7. Conclusions

1. A software implementation of the physics-informed neural network method has been developed. The corresponding classes of the definition of non-networks and the computational method make it possible to set a differential equation with boundary conditions and to determine the parameters when solving an inverse problem.

2. The method of neural networks has been used to solve inverse linear and nonlinear problems related to the mechanics of elastic bodies. It should be noted that the transition from linear problems to a geometrically nonlinear statement did not require a significant change in the architecture of neural networks. The relative error in the predicted values of the unknown parameters is within 3 %. At the same time, neural networks of direct signal propagation were used with the number of internal layers from 3 to 8 and neurons from 8 to 16. Such a relatively simple architecture is important for ensuring high calculation speed.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study, as well as the results reported in this paper.

Funding

The study was carried out within the framework of research work 0122U001765 “Thermochemical pressing of special purpose materials”.

Data availability

The manuscript has related data in the data repository, namely, the software implementation of the proposed methods and the results of solving the given problems, at the link https://github.com/avk256/AutoPINN.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the current work.

References

1. Edwards, C. H., Penney, D. E., Calvis, D. T. (2014). Differential Equations and Boundary Value Problems: Computing and Modeling. Boston: Pearson, 797.
2. Pinder, G. F. (2018). Numerical Methods for Solving Partial Differential Equations. Wiley, 304.
3. Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., Yang, L. (2021). Physics-informed machine learning. Nature Reviews Physics, 3 (6), 422–440. https://doi.org/10.1038/s42254-021-00314-5
4. Willard, J., Jia, X., Xu, S., Steinbach, M., Kumar, V. (2022). Integrating Scientific Knowledge with Machine Learning for Engineering and Environmental Systems. ACM Computing Surveys, 55 (4), 1–37. https://doi.org/10.1145/3514228

5. Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems, 2 (4), 303–314. https://doi.org/10.1007/bf02551274
6. Lagaris, I. E., Likas, A., Fotiadis, D. I. (1998). Artificial neural networks for solving ordinary and partial differential equations. IEEE Transactions on Neural Networks, 9 (5), 987–1000. https://doi.org/10.1109/72.712178
7. Raissi, M., Perdikaris, P., Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045
8. Yarosh, A. O., Kudin, O. V. (2024). Neural network methods for solving elasticity problems. Visnyk of Kherson National Technical University, 1 (88), 295–305. https://doi.org/10.35546/kntu2078-4481.2024.1.41
9. Meethal, R. E., Kodakkal, A., Khalil, M., Ghantasala, A., Obst, B., Bletzinger, K.-U., W chner, R. (2023). Finite element method-enhanced neural network for forward and inverse problems. Advanced Modeling and Simulation in Engineering Sciences, 10 (1). https://doi.org/10.1186/s40323-023-00243-1
10. Lu, L., Meng, X., Mao, Z., Karniadakis, G. E. (2021). DeepXDE: A Deep Learning Library for Solving Differential Equations. SIAM Review, 63 (1), 208–228. https://doi.org/10.1137/19m1274067
11. Zhou, H., Pu, J., Chen, Y. (2023). Data-driven forward–inverse problems for the variable coefficients Hirota equation using deep learning method. Nonlinear Dynamics, 111 (16), 14667–14693. https://doi.org/10.1007/s11071-023-08641-1
12. Zhong, M., Yan, Z. (2023). Data-driven forward and inverse problems for chaotic and hyperchaotic dynamic systems based on two machine learning architectures. Physica D: Nonlinear Phenomena, 446, 133656. https://doi.org/10.1016/j.physd.2023.133656
13. Depina, I., Jain, S., Mar Valsson, S., Gotovac, H. (2021). Application of physics-informed neural networks to inverse problems in unsaturated groundwater flow. Georisk: Assessment and Management of Risk for Engineered Systems and Geohazards, 16 (1), 21–36. https://doi.org/10.1080/17499518.2021.1971251
14. Garay, J., Dunstan, J., Uribe, S., Costabal, F. S. (2023). Physics-informed neural networks for blood flow inverse problems. arXiv. https://doi.org/10.48550/arXiv.2308.00927
15. Mao, Z., Jagtap, A. D., Karniadakis, G. E. (2020). Physics-informed neural networks for high-speed flows. Computer Methods in Applied Mechanics and Engineering, 360, 112789. https://doi.org/10.1016/j.cma.2019.112789
16. Wang, Z. Q., Jiang, J., Tang, B. T., Zheng, W. (2014). High Precision Numerical Analysis of Nonlinear Beam Bending Problems under Large Deflection. Applied Mechanics and Materials, 638-640, 1705–1709. https://doi.org/10.4028/www.scientific.net/amm.638-640.1705
17. Wu, C., Zhu, M., Tan, Q., Kartha, Y., Lu, L. (2023). A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. Computer Methods in Applied Mechanics and Engineering, 403, 115671. https://doi.org/10.1016/j.cma.2022.115671
18. AutoPINN. Available at: https://github.com/avk256/AutoPINN
19. Segall, A. E. (2023). The Search for a Generalized Analytical Solution for the Inverse Problem; Some Surprisingly Simple Approximate Methods for Problems of Practical Importance. Journal of Physics: Conference Series, 2444 (1), 012013. https://doi.org/10.1088/1742-6596/2444/1/012013
20. Kern, M. (2016). Numerical Methods for Inverse Problems. Wiley. https://doi.org/10.1002/9781119136941