

THE DYNAMIC PROBLEM OF AN ELASTICALLY SUPPORTED BEAM ANALYSIS USING NEURAL ARCHITECTURE SEARCH WITH AUTOMATIC DIFFERENTIATION

**Konstantinos S. Demertzis¹, Konstantinos E. Morfidis², Konstantinos G. Kostinakis³,
Lazaros S. Iliadis⁴**

¹ School of Science & Technology, Informatics Studies, Hellenic Open University, Greece
e-mail: demertzis.konstantinos@ac.eap.gr

² Earthquake Planning and Protection Organization (EPPO-ITSK), Terma Dasylliou, 55535, Thessaloniki, Greece
e-mail: konmorf@gmail.com

³ Department of Civil Engineering, Aristotle University of Thessaloniki, Aristotle University Campus, 54124, Thessaloniki, Greece
e-mail: kkostina@civil.auth.gr

⁴ School of Engineering, Department of Civil Engineering, Faculty of Mathematics Programming and General Courses, Democritus University of Thrace, Kimmeria, Xanthi, Greece
e-mail: liiadis@civil.duth.gr

Abstract

The equations of motion for linear structures considered as continuous systems are generally non-homogeneous linear partial differential equations. Their solution is functions of time and spatial variables. The traditional analysis approach involves solving the eigenvalue problem. This yields the eigenvalues and eigenvectors, which can be used to construct the general solution of the differential equations. This approach has been widely used in mechanics and has proven effective in solving many engineering problems. In recent years, machine learning methods, particularly neural networks, have also been used to solve differential equations. These methods can learn the equation's solution directly from data without the need for explicit analytical expressions. This makes them particularly useful for complex problems that may need analytical solutions or for problems where obtaining analytical solutions takes time and effort. This paper compares the performance of these two approaches for a specific problem of the vibration of the Euler-Bernoulli beam on a Winkler-type elastic foundation subjected to a moving load. The machine learning approach learns the solution directly from data generated by solving the differential equation using a Neural Architecture Search (NAS) with Automatic Differentiation (AD) method (NASAD). The paper provides insights into the relative strengths and weaknesses of each method and highlights the potential of machine learning to solve complex problems with high accuracy and low computational sources.

Keywords: Neural Architecture Search; Automatic differentiation; Dynamic response; Beams on elastic foundation; Eigenvalue problem; Forced vibrations

1 INTRODUCTION

The solution of the equation of motion of structures is one of the most important issues in earthquake engineering as it is directly related to the determination of their seismic response [1]. The equations governing the motion are generally non-homogeneous linear partial differential equations with respect to time and spatial variables. This is due to the fact that structures are in effect “continuous systems” with infinite degrees of freedom [2]. Solving equations of this type is particularly difficult or even impossible as structures consist of many more than one structural element. Thus, for the solution of the equations of dynamic equilibrium the consideration of structures as discrete systems has prevailed [1]. In this case the structures are transformed into systems with a finite number of degrees of freedom by means of inertial and elastic discretization (“discrete systems”). Thus, in combination with the application of the finite element method, the solution of the dynamic problem is achieved by solving systems of second-order linear differential equations for which there is a large number of implementations available in computer code (see e.g. [3]). Despite the fact that nowadays the Finite Element Method is almost exclusively used for solving dynamic problems of structures (see e.g. [4,5,6]), the study of the solution in terms of “continuous systems” remains particularly useful for applications to simple structural systems such as elastically supported beams which can also be used for example to model pile foundation systems (see e.g. [7,8,9]).

The difficulty of solving the dynamic problem by considering structures as “continuous systems” (as described above) leads to the consideration of the possible feasibility of solving it using methods that fall within the scientific field of machine learning. This paper proposes a method to learn the solution directly from data generated by solving the differential equation using a neural architecture search with automatic differentiation. Automatic Differentiation (AD) [10] is a technique used to compute the derivatives of a function concerning its inputs and is a crucial component of modern neural network training algorithms [11]. On the other hand, the Neural Architecture Search (NAS) [12] is a technique used in machine learning to design and optimize neural network architectures automatically [13]. In traditional machine learning approaches, the architecture of a neural network is often manually designed by a human expert, which can be a time-consuming and challenging process. NAS automates this process to search for the optimal neural network architecture based on predefined constraints or objectives. The process involves exploring a large search space of possible network architectures and evaluating their performance on a given task. To apply the Neural Architecture Search with Automatic Differentiation (NASAD) method as a technique for the solution of differential equations, we formulate the problem as an optimization problem, where the goal is to minimize the error between the predicted and actual solution of the differential equation. The neural network architecture is designed to take the initial conditions and any relevant differential equation parameters as inputs and output the solution at a given time. The optimization problem is solved using gradient descent, where the error gradient with respect to the network weights is computed using automatic differentiation [14]. This allows the network weights to be updated to minimize the error between the predicted and actual solution of the differential equation.

The proposed method is a pilot application to optimal solution of dynamic problems of structures considered as continuous systems. To this end, the solution of the dynamic problem of the Euler-Bernoulli beam supported on a Winkler-type elastic foundation, which has been studied in a large number of published research papers using various classical methods of structural dynamics (see e.g. [15,16,17,18]), has been chosen at this stage of the investigation. More specifically, in this paper the problem of vibration of elastically supported Euler-Bernoulli beams due to a moving concentrated force (see e.g. [19,20,21]) is studied. The problem is solved both by the classical method of separation of variables and application of the eigenproblem solution

[1,2,22] and by NASAD method. The results of the solutions by both methods prove that the NASAD method can approach the results obtained by the classical solution method to an extremely high degree. This conclusion sets the stage for further investigation of the application of machine learning methods to the solution of dynamic equilibrium problems of more complex structures.

2 THE SOLUTION OF THE EQUATION OF MOTION FOR EULER-BERNOULLI BEAMS ON WINKLER TYPE ELASTIC FOUNDATION

2.1. The solution procedure based on the eigenvalue analysis approach

The governing equation of motion for Euler-Bernoulli beams on a Winkler type elastic foundation (ignoring the damping) in the framework of the first order theory is [23]:

$$EI \cdot \frac{\partial^4 w}{\partial x^4} + \rho A \cdot \frac{\partial^2 w}{\partial t^2} + k \cdot w = p(x, t) \quad (1)$$

Where:

$w(x, t)$ is the lateral deflection of cross-sectional centroid axis (m),

EI is the flexural stiffness of beam (kNm^2),

ρ is the mass density (mass per unit of area, (kg/m^2)),

A is the cross-sectional area in (m^2),

k is the modulus of sub-grade reaction (kN/m^2),

$p(x, t)$ is the external dynamic load (kN/m)

If the external dynamic load $p(x, t)$ is equal to zero then the equation (1) governs the free vibration of a beam without damping. In this case the vibration is due to the initial conditions which regard the imposed displacements and velocities at $t=0$.

A conventional procedure for the solution of (1) (with or without external dynamic load) is the mode superposition method [1,2]. In the framework of this method the eigenmodes of the vibrating structural system must be calculated at first. To this end, the eigenvalue problem must be solved. This problem is governed by Eq. (1) ignoring the external load (i.e. $p(x, t)=0$). The solution is based on the hypothesis that the unknown function of lateral deflection $w(x, t)$ can be expressed as a product of an unknown function $X(x)$ of the spatial variable x and an also unknown function $f(t)$ of time t (method of separation of variables or Fourier method). Thus:

$$w(x, t) = X(x) \cdot f(t) \quad (2)$$

Substituting the Eq. (2) in Eq. (1) and after the appropriate transformations the two following differential equations are extracted:

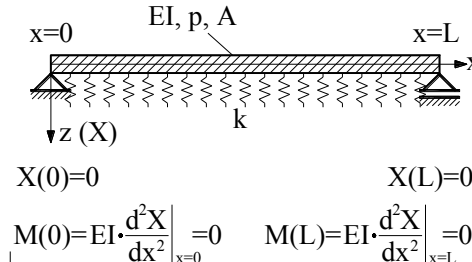
$$EI \cdot \frac{d^4 X}{dx^4} + (k - \omega^2 \cdot \rho A) \cdot X(x) = 0 \Rightarrow \frac{d^4 X}{dx^4} - \frac{(\omega^2 \cdot \rho A - k)}{EI} \cdot X(x) = 0 \quad (3a)$$

$$\frac{d^2 f}{dt^2} + \omega^2 \cdot f(t) = 0 \quad (3b)$$

Where $X(x)$ and ω are an eigenvector and its corresponding eigenfrequency.

The Eq. (3a) is a linear homogeneous differential equation with constant coefficients which can be solved using four boundary conditions that are depended on the types of the supports of the beam (for a simply supported beam: Fig. 1). The general form of the solution of (3a) is:

$$X(x) = C_1 \cdot e^{PL} + C_2 \cdot e^{-PL} + C_3 \cdot \cos(PL) + C_4 \cdot \sin(PL) \quad \text{where} \quad P = P(\omega) = \sqrt[4]{\frac{\omega^2 \cdot \rho A - k}{EI}} \quad (4)$$



$$\left\{ \begin{array}{l} X(0)=0 \\ \frac{d^2X}{dx^2}\bigg|_{x=0}=0 \\ X(L)=0 \\ \frac{d^2X}{dx^2}\bigg|_{x=L}=0 \end{array} \right\} \Rightarrow \begin{bmatrix} 1 & 1 & 1 & 0 \\ P^2 & P^2 & -P^2 & 0 \\ e^{PL} & e^{-PL} & \cos(PL) & \sin(PL) \\ P^2 e^{PL} & P^2 e^{-PL} & -P^2 \cos(PL) & -P^2 \sin(PL) \end{bmatrix} \cdot \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$\mathbf{D}(P(\omega)) \cdot \mathbf{C} = \mathbf{0}$

Boundary conditions $\det[\mathbf{D}(P(\omega))]=0 \Rightarrow$ Eigenfrequencies $\omega_i \Rightarrow$ Eigenmodes $X_i(x)$

Figure 1: Definition and solution of the eigenvalue problem of a simply supported Euler beam on elastic foundation

The calculation of the constants of integration C_1 - C_4 can be achieved using the four boundary conditions which form a homogeneous system with four equations. The non-trivial solution of this system arises when the determinant of the matrix \mathbf{D} of the coefficients C_1 - C_4 (Fig. 1) is set to zero:

$$\det[\mathbf{D}(P(\omega))]=0 \quad (5)$$

The Eq. (5) is the characteristic equation of the eigenvalue problem. It can be proved that for the simply supported beam of Fig. 1 the characteristic equation is:

$$\det[\mathbf{D}(P(\omega))]=0 \Rightarrow \sin\left[\left(\sqrt[4]{\frac{\omega^2 \cdot \rho A - k}{EI}}\right) \cdot L\right] = 0 \Rightarrow \omega_i = \sqrt{\frac{1}{\rho A} \cdot \left[\frac{i^4 \cdot \pi^4 \cdot EI}{L^4} + k\right]} \quad (i=1,2,\dots) \quad (6)$$

This trigonometric equation has infinite solutions which are the eigenvalues ω_i of the vibrating system. For each one of the eigenvalues ω_i the Eq. (4) leads to an eigenfunction $X_i(x)$. For the simply supported beam of Fig. 1 the form of the eigenfunctions is:

$$X_i(x) = \sin(P_i x) \quad \text{with} \quad P_i = P(\omega_i) = \sqrt[4]{\frac{\omega_i^2 \cdot \rho A - k}{EI}} \quad (7)$$

Once the eigenfrequencies and the corresponding eigenfunctions are known, the solution of (1) for the undamped free vibration ($p(x,t)=0$) can be expressed in a infinite series of terms:

$$w(x,t) = \sum_{i=1}^{\infty} X_i(x) \cdot q_i(t) \quad (8)$$

Thus, the solution of Eq. (1) (for $p(x,t)=0$) is based on the complete set of the eigenmodes of the studied beam and on the “main coordinates” $q_i(t)$. In other words, the solution (8) i.e. the unknown function of the lateral deflection $w(x,t)$ is expressed by means of the eigenvectors. The participation of each one of the eigenvectors to the solution is represented by the functions $q_i(t)$ as it is shown in Fig. 2.

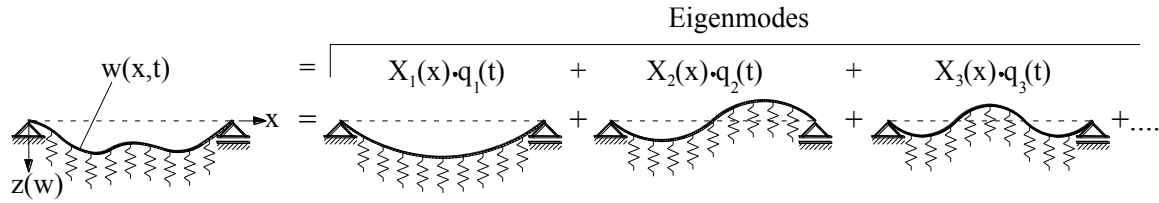


Figure 2: The rationale of the modal superposition method

In many cases an acceptable solution (as regards the precision) is reachable considering only a finite number of terms in (8).

Let's now consider the case of the vibration of a simply supported beam on a Winkler-type elastic foundation due to a moving transverse load P_e which is moved on the beam with velocity equal to c (Fig. 3).

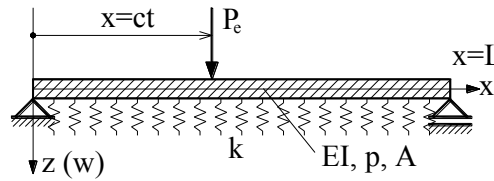


Figure 3: The vibration of a simply supported beam on a Winkler-type elastic foundation due to a moving transverse load

For this case the external dynamic load $p(x,t)$ in the Eq. 1 is expressed as follows:

$$p(x,t) = P_e \cdot \delta(x - c \cdot t) \quad (9)$$

Where δ is the Dirac function, whereas c is the velocity (m/sec) of the external load P_e .

The substitution of Eq. (8) to Eq. (1) leads to the following Equation:

$$\sum_{i=1}^{\infty} \left[q_i(t) \cdot EI \cdot \frac{d^4 X_i(x)}{dx^4} \right] + \sum_{i=1}^{\infty} \left[\rho A \cdot \frac{d^2 q_i(t)}{dt^2} \cdot X_i(x) \right] + k \cdot \sum_{i=1}^{\infty} [q_i(t) \cdot X_i(x)] = P_e \cdot \delta(x - c \cdot t) \quad (10)$$

Then, an infinite sequence of differential equations is created by means of manipulation of (10) with each one of the infinite eigenmodes $X_k(x)$ and the integration along the length L of the beam.

$$\sum_{i=1}^{\infty} \left[q_i \cdot \int_0^L EI \cdot \frac{d^4 X_i}{dx^4} \cdot X_k \cdot dx \right] + \sum_{i=1}^{\infty} \left[\frac{d^2 q_i}{dt^2} \cdot \int_0^L \rho A \cdot X_i \cdot X_k \cdot dx \right] + k \cdot \sum_{i=1}^{\infty} \left[q_i \cdot \int_0^L X_i \cdot X_k \cdot dx \right] = \int_0^L X_k \cdot P_e \cdot \delta(x - c \cdot t) \cdot dx \quad (11)$$

Using the orthogonality conditions:

$$\int_0^L EI \cdot \frac{d^4 X_i}{dx^4} \cdot X_k \cdot dx = \omega_i^2 \cdot \rho A \cdot N_i \cdot \delta_{ik} \quad \text{and} \quad \int_0^L \rho A \cdot X_i \cdot X_k \cdot dx = \rho A \cdot N_i \cdot \delta_{ik} \quad (12)$$

where: $N_i = \int_0^L X_i^2 \cdot dx = \frac{L}{2} - \frac{\sin(2 \cdot P_i \cdot L)}{4 \cdot P_i}$ and δ_{ik} = Kronecker delta

the infinite number of equations take the following general form:

$$\frac{d^2 q_i(t)}{dt^2} + \left(\frac{\omega_i^2 \cdot M_i + k \cdot N_i}{M_i} \right) \cdot q_i(t) = \left(\frac{P_e}{M_i} \right) \cdot \sin(P_i \cdot c \cdot t) \quad \text{where} \quad M_i = \rho A \cdot N_i \quad (13)$$

Using as initial conditions for the above equation:

$$q_i(t)=0 \quad \text{and} \quad \frac{dq_i(t)}{dt}=0 \quad (14)$$

the solution of (13) is:

$$q_i(t)=\frac{2 \cdot P_e}{L \cdot [k + \rho A \cdot (\omega_i^2 - c^2 \cdot P_i^2)]} \cdot \left[\sin(P_i \cdot c \cdot t) - \frac{P_i \cdot c}{\sqrt{\omega_i^2 + (k/\rho A)}} \cdot \sin\left(\sqrt{\frac{\omega_i^2 \cdot \rho A + k}{\rho A}} \cdot t\right) \right] \quad (15)$$

Combining the Eqs. (8), (15) and considering the Eq. (9) the form of the solution of Eq. (1) is:

$$w(x,t)=\sum_{i=1}^{\infty} X_i(x) \cdot \frac{2 \cdot P_e}{L \cdot [k + \rho A \cdot (\omega_i^2 - c^2 \cdot P_i^2)]} \cdot \left[\sin(P_i \cdot c \cdot t) - \frac{P_i \cdot c}{\sqrt{\omega_i^2 + (k/\rho A)}} \cdot \sin\left(\sqrt{\frac{\omega_i^2 \cdot \rho A + k}{\rho A}} \cdot t\right) \right] \quad (16)$$

2.2. The solution procedure based on the NASAD approach

In this paper, based on the meta-learning logic [24], we build a system for identifying the optimal hyperparameters of a neural network model using the NAS technique [25]. It is a technique to automate the design of artificial neural networks by searching for optimal hyperparameters, minimizing the number of operations required while providing an explicit methodology for knowledge discovery in unknown environments. Typical hyperparameters the technique in question can optimize include the optimization algorithms (SGD, Adam, etc.), learning rate, regularization, etc. Essentially, it enables the creation of optimal learning techniques for high-performance success with minimal up-front effort and minimal expertise.

In particular, and given a neural architecture search space F , where the input data D is divided into D_{train} and D_{val} and the cost function $Cost(\cdot)$ (e.g., accuracy, mean squared error, etc.), the goal is to find an optimal neural network $f^* \in F$ that can achieve the lowest cost on the dataset D . Finding the optimal neural network f^* is equivalent to [12,24,26]:

$$f^* = \operatorname{argmin}_{f \in F} Cost(f(\theta^*), D_{val}) \quad (17)$$

$$\theta^* = \operatorname{argmin}_{\theta} L(f(\theta), D_{train}) \quad (18)$$

Where θ^* is the learning parameter of the network.

The mode of operation of the proposed NAS strategy is enhanced by techniques based on how nature works and, in particular, finding analogies between techniques where instinct as a genetic trait overrides training. For example, some species in biology possess predatory behaviors from birth, allowing them to perform complex motor and sensory tasks without learning, which in most cases are fully sufficient for the species' survival. In contrast, in training artificial neurons it is usually chosen to perform a task, an architecture that is considered suitable for modeling the task; the search mainly focuses on finding the weight parameters using a learning algorithm. Inspired by social behaviors evolved in nature, neural networks can be developed with architectures that can perform a given task even when the weight parameters are randomized. Hence, they can perform well without training, and their performance can be further maximized through training.

Thus, the proposed architecture is characterized by a stacked hierarchy of layers, where at each time step t , the first recurrent layer is fed by the external input $u(t)$. In contrast, each successive layer is fed by the output of the previous one in the stack. Although their architectural organization allows for general flexibility in the size of each layer, to avoid complexity, we consider a hierarchical setup with recurrent layers N_L , each containing the same number of units

N_R . Moreover, we use $x^{(l)}(t) \in R^{N_R}$ to denote the state of layer l at time t . Omitting bias terms, the state transition function of the first layer is defined as follows [27,28]:

$$x^{(1)}(t) = (1 - \alpha^{(1)}) \cdot x^{(1)}(t-1) + \alpha^{(1)} \cdot \tanh(W_{in} \cdot u(t) + \hat{W}^{(1)} \cdot x^{(1)}(t-1)) \quad (19)$$

For each layer $l > 1$:

$$x^{(l)}(t) = (1 - \alpha^{(l)}) \cdot x^{(l)}(t-1) + \alpha^{(l)} \cdot \tanh(W^{(l)} \cdot u^{(l-1)}(t) + \hat{W}^{(l)} \cdot x^{(l)}(t-1)) \quad (20)$$

where $W_{in} \in R^{N_R \times N_U}$ is the input weight matrix, $\hat{W}^{(l)} \in R^{N_R \times N_R}$ is the recurrence weight matrix for level l , $W^{(l)} \in R^{N_R \times N_R}$ is the matrix concerning the connection weights between levels from level $l-1$ to level l , $\alpha^{(l)}$ is the leaky parameter at level l . Finally, \tanh represents the elementary application of the hyperbolic tangent.

Random weights improve the generalization properties of the solution of an original linear system because they produce nearly orthogonal (weakly correlated) features. Given that the output of the linear system is always correlated with the input data if the range of solution weights is limited, orthogonal inputs provide a wider range of solutions than those supported by weights. Also, small variations in weights allow the system to become more stable and noise tolerant, as input errors will not be amplified at the output of a linear system with little correlation between input and output weights. Thus, the random ranking of weights, which produces weakly correlated features in the hidden layer, allows us a satisfactory solution and good generalization performance. In general, it is an evolutionary neural network development strategy that can perform a specialized task independently of the connection weights, equivalent to the absence of training. The logic of not using training involves a basic exploration in the search for neural network architectures with specific biases that can potentially perform categorization on the given problem, even when using random weights. By exploring such architectures, it is possible to explore agents that can perform well in their interaction environment without the need to be trained. An engineering system can create robust self-determining systems capable of coping with complex situations.

To evaluate and validate the final model capability resulting from applying the proposed methodology, we use the [29,30,31] constant, which allows us to study the behavior of the scattering transformation when a set of similar inputs is introduced as input. This transformation can approximate the operation of a simple neural network architecture by allowing the study of how neural networks succeed in solving difficult problems in which multiscale feature extraction is required. At the same time, the properties of the transformation in question explain how a neural network can achieve invariance to input displacement and small input deformations, such as in cases of elastic deformation. In particular, new inputs are generated when a very small variation p is added to the input h , so that we obtain the new input $h + p$, which, with an appropriately chosen input function p , is ordered differently from the original input such that [32,33]:

$$\|S[m](h+p) - S[m](h)\| \leq \|p\| \quad (21)$$

Thus, it follows that the output for a new variable input does not differ from the original input by more than $\|p\|$. So, if the transformation follows the constraints of the scattering transformation:

$$\sum_{i=1}^N |\hat{\psi}_{(i,j)(\omega)}|^2 \leq \frac{C^2}{N}, \quad |\hat{\phi}_{(\omega)}|^2 \leq C^2 \quad (22)$$

For $C \in R$:

$$\|S[m](h+p) - S[m](h)\| \leq C^{m+1} \|p\| \quad (23)$$

This means that the constant C determines how vulnerable the transformation is to changes in the input by p . Therefore, as the Lipschitz constant determines the ability of the classifier to cope with new inputs, in the considered research, we propose to use it to detect how this constant evolves when searching for hyperparameters of a neural network. In particular, suppose that the input of a convolutional neural network is in the form of a vector, with $f(x_{in}, c)$ the network output for the category c with the vector x_{in} as input. Suppose two different input vectors y_{in}, h_{in} and the corresponding outputs $f(y_{in}, c), f(h_{in}, c)$ so as y_{ik}, h_{ik} the k -th level outputs in the channel i for each of the two inputs. The convolutional neural network consists of convolution layers, pooling layers, and ReLU activation functions. So, for each of the three kinds of layers, we have:

- 1) Let layer k is a convolution layer. As we express the inputs as one-dimensional vectors, the convolution with a two-dimensional kernel ψ_{ijk} , which connects the i channel of the output to the j channel of the input, is implemented by multiplying the input vector by a matrix A_{ijk} generated by the original kernel, such that [30,34]:

$$x_{ik} = \sum_{j=1}^{N_k} A_{ijk} \cdot x_{j(k-1)} \quad i=1,2,\dots,M_k \quad (24)$$

where N_k is the number of channels of the input and M_k is the number of channels of the output of the convolutional level k . Thus:

$$\begin{aligned} \|y_{ik} - h_{ik}\|_2 &= \left\| \sum_{j=1}^{N_k} A_{ijk} \cdot y_{j(k-1)} - \sum_{j=1}^{N_k} A_{ijk} \cdot h_{j(k-1)} \right\|_2 = \left\| \sum_{j=1}^{N_k} A_{ijk} \cdot (y_{j(k-1)} - h_{j(k-1)}) \right\|_2 \\ &\leq \sum_{j=1}^{N_k} \|A_{ijk} \cdot (y_{j(k-1)} - h_{j(k-1)})\|_2 \leq \sum_{j=1}^{N_k} \|A_{ijk}\| \cdot \|y_{j(k-1)} - h_{j(k-1)}\|_2 \\ &\Rightarrow \|y_{ik} - h_{ik}\|_2 \leq \sum_{j=1}^{N_k} \|A_{ijk}\| \cdot \|y_{j(k-1)} - h_{j(k-1)}\|_2 \end{aligned} \quad (25)$$

Suppose that k is a Pooling plane in which no overlapping regions exist, then:

$$\|y_{ik} - h_{ik}\|_2 \leq \|y_{j(k-1)} - h_{j(k-1)}\|_2 \quad (26)$$

- 2) Assuming that k is ReLU, then the output vector has the form:

$$x_{ik} = [x_{ik}(1) \quad x_{ik}(2) \quad \dots \quad x_{ik}(m)]^T \quad (27)$$

The output $x_{ik}(t)$ defined as:

$$x_{ik}(t) = \max(0, x_{i(k-1)}(t)) \quad (28)$$

Thus

$$\begin{aligned} \|y_{ik} - h_{ik}\|_2^2 &= \sum_{t=1}^m \left| \max(0, y_{i(k-1)}(t)) - \max(0, h_{i(k-1)}(t)) \right|^2 \\ &\leq \sum_{t=1}^m \left| y_{j(k-1)}(t) - h_{j(k-1)}(t) \right|^2 = \|y_{j(k-1)} - h_{j(k-1)}\|_2^2 \Rightarrow \|y_{jk} - h_{jk}\|_2 \\ &\leq \|y_{j(k-1)} - h_{j(k-1)}\|_2 \end{aligned} \quad (29)$$

where $|\max(0, \alpha) - \max(0, \beta)| \leq |\alpha - \beta|$.

Using the above equations, a constant L_{ik} can be calculated for which it holds that:

$$\|y_{jk} - h_{jk}\|_2 \leq L_{ik} \cdot \|y_{10} - h_{10}\|_2 \quad (30)$$

We define the constant recursively as $L_{ik} = 1$, so that for each type of level it holds:

- Convolution layer: $L_{ik} = \sum_{j=1}^{N_k} \|A_{ijk}\|_2 \cdot L_{j(k-1)}$
- Pooling layer: $L_{ik} = L_{i(k-1)}$
- ReLU function: $L_{ik} = L_{i(k-1)}$

Therefore, if the network has p levels we can find the Lipschitz constant that satisfies the relation:

$$\|f(y_{in}, c) - f(h_{in}, c)\|_2 \leq L_{cp} \cdot \|y_{in} - h_{in}\|_2 \quad (31)$$

Having developed the method for finding a Lipschitz constant for the network, we study how it evolves when finding the appropriate hyperparameters of a neural network in solving the given problem.

AD is a technique used to compute derivatives of functions. It is commonly used in machine learning, optimization, and scientific computing. The basic idea of AD is to decompose a function into a sequence of elementary operations, such as addition, multiplication, and exponentiation, and then compute the function's derivative by applying the chain rule of calculus to each operation. This process can be done efficiently using computer algorithms without explicitly computing the function's derivative.

There are two main approaches to AD: forward mode and reverse mode. In forward mode, the derivatives of all intermediate variables are computed simultaneously as the function is evaluated. Specifically, forward accumulation computes the recursive relation [10,14]:

$$\frac{\partial w_i}{\partial x} = \frac{\partial w_i}{\partial w_{i-1}} \cdot \frac{\partial w_{i-1}}{\partial x} \quad \text{with } w=y \quad (32)$$

In contrast, in reverse mode, the derivatives of the output variable are computed first and then propagated backward through the sequence of operations. Specifically, reverse accumulation computes the recursive relation:

$$\frac{\partial y}{\partial w_i} = \frac{\partial y}{\partial w_{i+1}} \cdot \frac{\partial w_{i+1}}{\partial w_i} \quad \text{with } w=x \quad (33)$$

AD has several advantages over traditional methods of computing derivatives, such as finite differences and symbolic differentiation. It is more accurate and efficient, particularly for functions with a large number of variables or complex compositions of functions. It can also handle functions with discontinuities, such as piecewise-defined functions, more easily than other methods.

In this study, differential equations are solved using a neural search architecture with automatic differentiation that is trained to understand the behavior of the differential equation. The loss function for the network's training is the differential equation itself, which is then automatically differentiated to minimize. Below is a summary of the architecture at a high level:

1. The beginning and boundary conditions, as well as the differential equation to be solved, were defined.
2. A collection of input-output pairs was created from the differential equation, where the inputs represent the independent variables and the outputs represent the dependent variables.
3. A neural network was trained to forecast the values of the output given the values of the input.
4. The gradients of the loss function were computed with respect to the network parameters using automatic differentiation.
5. An optimizer was used to make network parameter updates that would reduce loss.

By minimizing the error between the predicted and actual solutions of the differential equation, the neural network can learn the system's underlying behavior and provide accurate solutions for different inputs. This approach can be particularly useful for problems where analytical solutions are not readily available or where the analytical solution is too complex to be obtained or is computationally expensive.

3 NUMERICAL EXAMPLE

This section presents the results of the comparative solution of the two methods presented in the previous section using the numerical example depicted in the Fig. 4.

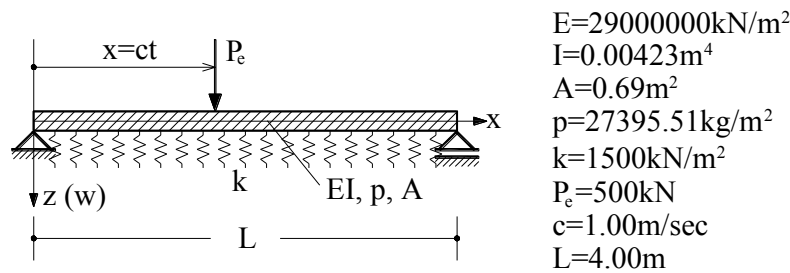


Figure 4: The values of the parameters of the studied numerical example

As far as the solution by the classical method of solving the differential equation of the problem (Eq. (1)) is concerned, it should be noted that only the first term in the equation (Eq. (16)) was used as the other terms turned out to have negligible influence. Thus, the solution of the problem by the classical method is obtained for the numerical example by the Eq. (34):

$$w(x,t)=X_1(x) \cdot \frac{2 \cdot P_e}{L \cdot [k + \rho A \cdot (\omega_1^2 - c^2 \cdot P_1^2)]} \cdot \left[\sin(P_1 \cdot c \cdot t) - \frac{P_1 \cdot c}{\sqrt{\omega_1^2 + (k/\rho A)}} \cdot \sin\left(\sqrt{\frac{\omega_1^2 \cdot \rho A + k}{\rho A}} \cdot t\right) \right] \quad (34)$$

The solution of the eigenproblem resulted in:

$$\omega_1 = 1.5964 \text{ rad/sec} \rightarrow P_1 = 0.7854 \rightarrow X_1(x) = \sin(0.7854 \cdot x) \quad (35)$$

As a parameter for comparing the results, the variation of the lateral deflection in the middle of the beam $w(L/2)$ as a function of time (t) was used. This variation is depicted in Fig. 5. It should be stressed that the solution resulting from Eq. (34) is valid for the time interval from $t=0$ sec to $t=L/c=4.00/1.00=4$ sec. For $t>4.00$ sec the moving load is outside the beam and the vibration becomes free. For this reason, the diagram in Fig. 5 has a time range of $[0.00\text{sec}, 4.00\text{sec}]$.

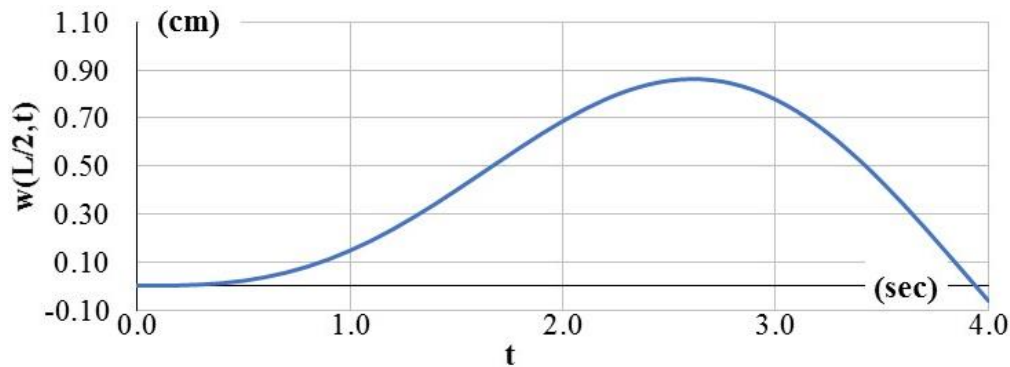


Figure 5: Time variation of the lateral deflection in the middle of the beam

Table 1 shows the results of the NASAD approach for predicting the displacement of the Euler-Bernoulli beam on an elastic foundation subjected to a moving load for 14 different time inputs. The table includes the time, the displacement calculated using the classical method and the displacement predicted by the NASAD approach.

ID	Time (sec)	Displacement (cm) – Classical method	Displacement (cm) - Predicted
1	0.3	0.004605411	0.004461994
2	0.63	0.040570695	0.039418149
3	0.65	0.044371532	0.042154818
4	0.66	0.046351195	0.0485189
5	0.67	0.048384207	0.051071586
6	2.15	0.760442817	0.758433355
7	2.44	0.85035003	0.849373095
8	2.6	0.865912826	0.865749313
9	2.86	0.830896232	0.832665859
10	3.27	0.621689162	0.624985816
11	3.46	0.469491684	0.474114383
12	3.6	0.340983069	0.350375689
13	3.77	0.172549928	0.177670435
14	3.95	-0.012383849	-0.008466525

Table 1: Comparative results extracted from the classical method and from the NASAD approach

The NASAD approach provides accurate displacement predictions for different time inputs, with small differences between the predicted and the classically calculated values.

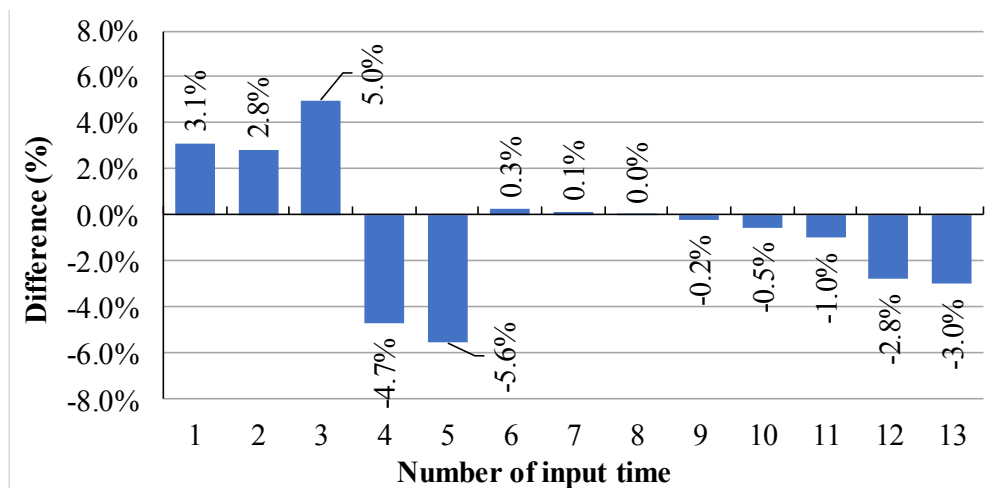


Figure 6: Differences (%) of classically calculated and predicted values (NASAD) for the input times of Table 1

Figure 6 depicts the comparison of the analytically calculated vs predicted displacements in terms of Difference (%). This comparison provides a visual representation of the accuracy of the NASAD approach in predicting the displacement of the Euler-Bernoulli beam on an elastic foundation subjected to a moving load. The predicted displacement is very close to the analytically calculated one, and the differences between the two methods are negligible, indicating the good performance of the NASAD approach. By comparing the analytically calculated and predicted displacement on the graph, the reader can understand the accuracy of the NASAD approach in predicting the displacement for different time inputs.

Training Set Mean Squared Error:	1.175e-06
Test Set Mean Squared Error:	8.422e-06
Mean Absolute Error:	0.00217
Root Mean Squared Error:	0.0029
Coefficient of Determination (R^2):	0.999965

Table 2: NASAD Performance Metrics

Table 2 provides the performance metrics for the NASAD approach used in this study to solve the differential equation. The metrics [35] include the Mean Squared Error (MSE) for the training and test sets, the Mean Absolute Error (MAE), the Root Mean Squared Error (RMSE), and the Coefficient of Determination (R^2). The training set MSE is 1.175e-06, indicating that the predicted values are very close to the actual values in the training set. The test set MSE is 8.422e-06, which is slightly higher than the training set MSE, but still a relatively small value, indicating the model's good performance on unseen data. The MAE is 0.00217, which measures the average magnitude of the errors in the predictions. The RMSE is 0.0029, which measures the average magnitude of the errors in the predictions, with higher weight given to larger errors. These values suggest that the NASAD approach has low errors and provides accurate predictions. Finally, the R^2 value is 0.999965, which measures the model's goodness of fit. This value is very close to 1, indicating that the NASAD approach provides an excellent fit to the data, with only a small amount of unexplained variance. Overall, these performance metrics suggest that the NASAD approach effectively solves the differential equation and provides accurate predictions of the displacement of the Euler-Bernoulli beam on an elastic foundation subjected to a moving load.

4 CONCLUSIONS

The solution of dynamic problems of structures is a significant field of research in the context of earthquake engineering. The solution of these problems by considering structures as "continuous systems" is very difficult or even impossible, except for very simple problems of structural systems consisting of a single structural member. Thus, for the easy and sufficiently accurate solution of the dynamic problems of structures, extensive use is made of discrete systems obtained by elastic and inertial discretization. The discrete systems lead to reliable results using the finite element method.

The present paper proposes a different approach to solving the dynamic problem of continuous systems by applying machine learning methods and, more specifically, the NASAD method based on artificial neural networks and automatic differentiation. The paper aims to document that these methods lead accurately to the solution of the differential equations governing the dynamic equilibrium of continuous systems. To this end, both a classical solution method (method of separation of variables and the solution of the eigenproblem) and the NASAD method have been used to study the vibration of an Euler-Bernoulli beam resting on

an elastic Winkler-type foundation subjected to a moving concentrated load. The traditional modal analysis approach is a well-documented method in mechanics for solving differential equations, but it is computationally intensive, especially for large and complex structures. On the other hand, the proposed machine learning method offers an alternative high-accurate approach that learns the solution directly from the data, which can be particularly useful for problems where obtaining analytical solutions requires time and effort. The comparative results of the two approaches showed that the considered NASAD method can lead to almost identical results with the classical solution method. In the future, the research will focus on how the proposed machine learning approach solves more complex structural systems.

Overall, this paper highlights the potential of machine learning methods in solving complex problems in engineering and mechanics and how they can complement traditional analytical methods. It's exciting to see how machine learning can advance our understanding and ability to solve complex problems in various fields.

REFERENCES

- [1] A.K. Chopra, *Dynamics of Structures, Theory and Applications to Earthquake Engineering*, 5th Edition, NJ, USA, Prentice Hall, 2016.
- [2] S.S. Rao, *Vibration of Continuous Systems*, Wiley, New York, 2007.
- [3] P. Bhatt, *Programming the Dynamic Analysis of Structures*, Spon Press, New York, 2002.
- [4] T.J.R. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Prentice-Hall International Editions, Inc., 1987.
- [5] K-J Bathe, *Finite Element procedures*, Prentice-Hall, Boston, 2006.
- [6] W. Weaver, P.R. Jonhston, *Structural Dynamics by Finite Elements*, 1st Edition, Prentice-Hall, Englewood Cliffs, NJ, 1987.
- [7] A.J. Valsangkar, R.B. Pradhanang, Free Vibration of Partially Supported Piles, *Journal of Engineering Mechanics*, **113(8)**, 1244-1247, 1987.
- [8] B.K. Lee, J.S. Jeong, L.G. Fan, T.K. Jin, Free Vibrations of Tapered Piles Embedded Partially in Winkler Type Foundations, *KSCE Journal of Civil Engineering*, **3(2)**, 195-203, 1999.
- [9] H.H. Catal, Free vibration of partially supported piles with the effects of bending moment, axial and shear force, *Engineering Structures*, **24**, 1615-1622, 2002.
- [10] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey." arXiv, Feb. 05, 2018. doi: 10.48550/arXiv.1502.05767.
- [11] J. Bolte and E. Pauwels, "A mathematical model for automatic differentiation in machine learning." arXiv, Oct. 29, 2020. doi: 10.48550/arXiv.2006.02080.
- [12] T. Elsken, J. H. Metzen, and F. Hutter, "Neural Architecture Search: A Survey." arXiv, Apr. 26, 2019. doi: 10.48550/arXiv.1808.05377.
- [13] G. Franchini *et al.*, "Neural architecture search via standard machine learning methodologies," *Math. Eng.*, vol. 5, no. 1, Art. no. mine-05-01-012, 2023, doi: 10.3934/mine.2023012.

- [14] W. Morningstar, S. Vikram, C. Ham, A. Gallagher, and J. Dillon, “Automatic Differentiation Variational Inference with Mixtures,” in Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, Mar. 2021, pp. 3250–3258. Accessed: Mar. 18, 2023. [Online]. Available: <https://proceedings.mlr.press/v130/morningstar21b.html>
- [15] M. Eisemberger, D.Z. Yankelevski, M.A. Adin, Vibration of beams fully or partially supported on elastic foundation, *Earthquake Engineering and Structural Dynamics*, **13**, 651 – 660, 1985.
- [16] F.W. Williams, D. Kennedy, Exact dynamic member stiffnesses for a beam on an elastic foundation, *Earthquake Engineering and Structural Dynamics*, **15**, 133-136, 1987.
- [17] Y.C. Lai, B.Y. Ting, W-S Lee, B.R. Becker, Dynamic Response of Beams on Elastic Foundation, *Journal of Structural Engineering*, **118(3)**, 853-858, 1992.
- [18] C-N. Chen, Vibration of Prismatic beam on an Elastic Foundation by differential quadrature element method, *Computers and Structures*, **77(1)**, 1-9, 2000.
- [19] L. Fryba, *Vibration of Solids and Structures under Moving Loads*, 3rd Edition, Thomas Telford, 1999.
- [20] O.R. Jaiswal, R.N. Iyengar, Dynamic response of a beam on elastic foundation of finite depth under a moving force, *Acta Mechanica*, **96**, 67-83, 1993.
- [21] W. Abbas, O.K. bakr, M.M. Nassar, M.A.M. Abdeen, M. Shabrawy, Analysis of Tapered Timoshenko and Euler-Bernoulli Beams on an Elastic Foundation with Moving Loads, *Journal of Mathematics*, Volume 2021, Article ID 6616707
- [22] R.W. Clough, J. Penzien, *Dynamics of Structures*, 3rd edition, Computers and Structures, Inc., 2003.
- [23] M. Rades, Steady-State Response of a finite beam on a Pasternak-Type Foundation, *International Journal of Solids and Structures*, **6**, 739 – 756, 1970.
- [24] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter, “Auto-Sklearn 2.0: Hands-free AutoML via Meta-Learning,” arXiv, arXiv:2007.04074, Sep. 2021. doi: 10.48550/arXiv.2007.04074.
- [25] M. Lindauer and F. Hutter, “Best Practices for Scientific Research on Neural Architecture Search.” arXiv, Nov. 03, 2020. doi: 10.48550/arXiv.1909.02453.
- [26] J. Casebeer, N. J. Bryan, and P. Smaragdis, “Meta-AF: Meta-Learning for Adaptive Filters.” arXiv, Nov. 21, 2022. doi: 10.48550/arXiv.2204.11942.
- [27] J.-S. Kang, J. Kang, J.-J. Kim, K.-W. Jeon, H.-J. Chung, and B.-H. Park, “Neural Architecture Search Survey: A Computer Vision Perspective,” *Sensors*, vol. 23, no. 3, Art. no. 3, Jan. 2023, doi: 10.3390/s23031713.
- [28] D. T. Speckhard, K. Misiunas, S. Perel, T. Zhu, S. Carlile, and M. Slaney, “Neural architecture search for energy-efficient always-on audio machine learning,” *Neural Comput. Appl.*, Feb. 2023, doi: 10.1007/s00521-023-08345-y.
- [29] G. Alessandrini, M. V. de Hoop, R. Gaburro, and E. Sincich, “Lipschitz stability for a piecewise linear Schrödinger potential from local Cauchy data,” *Asymptot. Anal.*, vol. 108, no. 3, pp. 115–149, Jan. 2018, doi: 10.3233/ASY-171457.

- [30] K. Demertzis, L. Iliadis, and P. Kikiras, “A Lipschitz - Shapley Explainable Defense Methodology Against Adversarial Attacks,” in *Artificial Intelligence Applications and Innovations. AIAI 2021 IFIP WG 12.5 International Workshops*, Cham, 2021, pp. 211–227. doi: 10.1007/978-3-030-79157-5_18.
- [31] C. Freer, B. rn Kjos-Hanssen, A. Nies, and F. Stephan, “Algorithmic Aspects of Lipschitz Functions,” *Computability*, vol. 3, no. 1, pp. 45–61, Jan. 2014, doi: 10.3233/COM-14025.
- [32] Y. Gao and L. Jia, “Stability in measure for uncertain delay differential equations based on new Lipschitz conditions,” *J. Intell. Fuzzy Syst.*, vol. 41, no. 2, pp. 2997–3009, Jan. 2021, doi: 10.3233/JIFS-210089.
- [33] K. Demertzis, L. Iliadis, P. Kikiras, and E. Pimenidis, “An explainable semi-personalized federated learning model,” *Integr. Comput.-Aided Eng.*, vol. Preprint, no. Preprint, pp. 1–16, Jan. 2022, doi: 10.3233/ICA-220683.
- [34] Y. Wang and M. Sui, “Finite lattice approximation of infinite lattice systems with delays and non-Lipschitz nonlinearities,” *Asymptot. Anal.*, vol. 106, no. 3–4, pp. 169–203, Jan. 2018, doi: 10.3233/ASY-171444.
- [35] A. Botchkarev, “Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology,” *Interdiscip. J. Inf. Knowl. Manag.*, vol. 14, pp. 045–076, 2019, doi: 10.28945/4184.