Optimization
Iranian Journal of Optimization
Volume 16, Issue 1, 2024, 73-87
Research Paper

Online version is available on: https://sanad.iau.ir/journal/ijo

Islamic Azad University
Rasht Branch
ISSN: 2588-5723
E-ISSN:2008-5427

Developing a Novel Neural Network Method for Solving Variable-Order Fractional Partial Differential Equations with Time-Varying Delay

Farahnaz Golpour Lasaki, Hamideh Ebrahimi* and Mousa Ilie

Department of Mathematics, Ra.C., Islamic Azad University, Rasht, Iran.

Revise Date: 16 March 2025
Accept Date: 04 June 2025

Abstract

This paper presents a novel functional link-based neural network designed to address a class of variable-order fractional partial differential equations with time-varying delay. Due to the proficiency of Lagrange polynomials in numerical approximations for fractional calculus, these polynomials serve as the foundational neuro-solutions within the neural network. Finding the right activation functions is essential for effective learning in artificial neural networks, particularly when solving variable-order fractional derivatives and time-varying delays. To reduce the computational complexity of the proposed neural network, a linear activation function is used. Numerical simulations are carried out to demonstrate the capability of the proposed method. The neural network undergoes training utilizing a modified Newton-Raphson approach rather than the traditional learning techniques. The study's findings indicate that the suggested functional link-based neural network achieves greater accuracy in comparison to some traditional methods for solving fractional partial differential equations

Keywords:
Functional Link-based Neural Network
Lagrange Polynomials
Variable-order Fractional Systems
Time-varying Delay

***Correspondence E-mail**: ebrahimi60@iau.ac.ir

INTRODUCTION

Fractional calculus, a branch of Applied Mathematics, focuses on non-integer order differentiation and integration. This discipline presents numerous advantages and has found applications in a wide range of scientific and engineering fields (Chávez-Vázquez et al., 2022; Diethelm et al., 2022). It provides a more accurate representation of complex systems, particularly those characterized by memory and hereditary effects, than traditional integer calculus. Additionally, fractional calculus enables the modeling and analysis of anomalous phenomena, allowing for a deeper understanding of intricate systems and their behavior.

Time-delay systems, often referred to as delay differential equations (DDEs), represent a distinct category of differential equations that incorporate time delays. These equations have been extensively utilized across diverse scientific domains, including economics, physics, ecology, and engineering control. While only a limited number of DDEs possess explicit analytical solutions, a surge in the development of numerical techniques has emerged recently to compensate for the existing gap in theoretical research. The study of delay fractional differential equations (DFDEs) has taken on increased significance in light of the growing relevance of fractional calculus. Prior research has introduced a range of both analytical and numerical methodologies to address DFDEs (Kumar & Erturk, 2022; Hattaf, 2022).

The dynamics described by variable-order fractional partial differential equations are finding increasing relevance in a variety of scientific and engineering domains. These include modeling the dynamics of the anomalous diffusion, synthesis of multifractional Gaussian noises, developing statistical mechanics models, and a predictive model of the spread of COVID-19 (Sheng et al., 2011; Singh et al., 2021). Deriving analytical solutions for problems that involve variable-order fractional derivatives poses significant challenges. Consequently, many researchers have turned to the development of numerical methods as viable alternatives. For example, a robust three-level explicit spline finite difference scheme has been proposed for a specific class of nonlinear time variable-order fractional partial differential equations (Moghaddam & Machado, 2017). Furthermore, a study in (Tayebi et al., 2017) explored a meshless approach to solve two-dimensional variable-order time-fractional advection-diffusion equations. Additionally, (Heydari & Avazzadeh, 2018). introduced the Legendre wavelets optimization technique for solving variable-order fractional Poisson equations.

While time-varying delay partial differential equations have attracted the attention of numerous researchers, there appears to be a limited exploration in the field of time-varying delay fractional partial differential equations with variable-order based on our current understanding. In this article, we introduce a neural network based model to solve the following category of variable-order fractional time-varying delay partial differential equations (VOFTVDPDEs). The equation is as follows

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \eta\frac{\partial^2 u(x,t)}{\partial x^2} = f\left(t, u(x,t), u\big(x, t - s(t)\big)\right),$$

$$a < x < b,\ 0 < t \le T, \tag{1}$$

$$u(x,t) = \psi(x,t),\ x \in (a,b),\ t \in [-s, 0], \tag{2}$$

$$u(a,t) = \psi(a,t),\ u(b,t) = \psi(b,t),\ t \in (0,T], \tag{3}$$

where $q(t)$, is the fractional variable-order derivative, $s(t) > 0$ is the time-varying delay, $s = \max_{0\le t\le T}\{s(t)\}$, $\psi(x,t)$ is a known function, and $0 < q(t) < 1$. f is a continuous function, $\eta > 0$ is the diffusion coefficient.

In the VOTVFDPDEs (1)-(3), both the order of the derivative and the time delay are functions of time, which can introduce additional complexity to the problem. Moreover, the VOTVFDDE (1)-(3) can be considered a broader framework encompassing some previously discussed research. For example, with considering variable-order $q(t)$ and time-varying delay $s(t) > 0$ as constant values, the VOTVFDPDEs (1)-(3) are reduced to fractional order partial differential equations with time delay .To the best of our knowledge, this is the first time that this problem

has been considered in the literature. Given the absence of an analytical solution, we aim to introduce a neural network (ANN) for solving the problem (1)-(3). ANNs, widely adopted in fractional calculus studies (Bijiga & Ibrahim, 2021). One of the primary advantages of employing ANN is the ability to acquire solutions in an analytical form, which is highly desirable in practical situations where continuous and differentiable functions are needed for a comprehensive study of solution properties. In addition to the analytical form of solutions, ANNs offer various other benefits. For instance, they consistently provide precise results throughout the entire domain, even when only a few data points are available. This enables researchers to effectively adjust the numerical technique based on initial conditions and simplifies the solving of complex problems due to the architecture and simplicity of ANN. Furthermore, ANNs prove to be valuable when there is a requirement to analytically integrate or differentiate the obtained solutions. By employing ANNs, one can conveniently perform these operations and study the properties of the solutions more effectively. ANNs are widely recognized and frequently used techniques in the field of machine learning. This is primarily due to their global approximation capability, which allows them to approximate and model various functions accurately. As a result, ANNs have gained significant popularity and find extensive applications in different fields.

To reference more recent works, refer to (Al-Janabi et al., 2020; Al-Janabi et al., 2020; Al-Janabi and Alkaim, 2020). Based on the previous discussion, we investigate the solutions of VOTVFDPDEs (1)-(3) through a novel class of artificial neural networks called functional link-based neural networks (FLNN). These networks offer advantages over multilayer neural networks (Pao, 1989). Unlike traditional neural networks, FLNNs do not include a hidden layer, resulting in a reduced number of unknown parameters compared to classical multilayer perceptrons (MLPs). Consequently, the computational burden is lessened with a single-layer FLNN structure. Recent studies have focused on classical polynomials within the FLNN framework (Zou et al., 2010; Peterson and Larin, 2008; Patra et al., 2008), particularly the Lagrange polynomial basis functions known for their rapid convergence and high accuracy. Hence, we propose a single-layer FLNN employing Lagrange polynomials to expand the input pattern for solving problem (1)-(3). By collocating the time domain, the VOTVFDPDEs problem can be transformed into an optimization problem, an essential aspect of neural network design (Mohammed & Al-Janabi, 2020; Kadhuim & Al-Janabi, 2023; Syah et al., 2022). Consequently, a backpropagation algorithm is applied to determine the unknown parameters of Lagrange functional link-based neural networks (LFLNN). Numerous iterative techniques, including gradient descent, Newton's method, and conjugate gradient, are available for this purpose. However, for the training of our neural network, we will utilize a modified version of the Newton-Raphson method and examine its convergence properties. The structure of the paper is as follows: Section 2 provides a review of the fundamental concepts of fractional variable-order operators and basis functions. The architecture of the LFLNN and the corresponding learning algorithm are detailed in Section 3. In order to assess the effectiveness of different activation functions and the proposed framework, several numerical examples are discussed in Section 4. We give the discussion of results and conclusion in Sections 5 and 6, respectively.

MATHEMATICAL PRELIMINARIES

In this section, we will present definitions related to variable-order calculus and basic functions, which will be utilized in the subsequent sections of the paper. For more on the subject, we refer the reader to (Patnaik et al., 2020).

Fractional variable-order calculus

Definition 2.1 *The Riemann-Liouville fractional variable-order integral is defined as follows*

$$ {}_0^C I_t^{q(t)} f(t) = \frac{1}{\Gamma(q(t))} \int_0^t (t-s)^{q(t)-1} f(s) ds, \quad 0 < q(t) < 1, \qquad (4) $$

where $f \in C^1(0,T)$.

Definition 2.2 *The Caputo fractional variable-order derivative is characterized by the following definition*

$$ {}_0^C D_t^{q(t)} f(t) = \frac{1}{\Gamma(1-q(t))} \int_0^t (t-s)^{-q(t)} f'(s) ds, \quad 0 < q(t) < 1, \tag{5}$$

where $f \in C^1(0,T)$.

Definition 2.3 *The Caputo fractional variable-order partial differential operator of order* $0 < q(t) < 1$ *with the lower limit* $t_0 = 0$ *is defined as* (Zúniga-Aguilar et al., 2019).

$$ \frac{\partial^{q(t)} u(x,t)}{\partial t^{q(t)}} = \begin{cases} \frac{1}{\Gamma(1-q(t))} \int_0^t (t-r)^{-q(t)} \frac{\partial u(x,r)}{\partial r} dr, & 0 < q(t) < 1, \\ \frac{\partial u(x,t)}{\partial t}, & q(t) = 1. \end{cases} \tag{6}$$

When the value of $q(t)$ is a fixed constant q, it results in the qth-order Caputo fractional derivative. It is feasible to compute the variable-order fractional derivative for polynomial expressions, especially when $0 < q(t) < 1$, as described by the following formula (Akgül et al., 2017).

$$ \frac{\partial^{q(t)} t^n g(x)}{\partial t^{q(t)}} = \begin{cases} 0, & n = 0, \\ \frac{\Gamma(n+1)}{\Gamma(n+1-q(t))} t^{n-q(t)} g(x), & n \in \mathbb{N}. \end{cases} \tag{7}$$

Legendre polynomials

Here, the foundation of our discussion includes an overview of Legendre polynomials. The basis functions of Legendre, applicable within the range $[-1,1]$, can be derived through the use of Legendre's differential equation as (Patra et al., 2008)

$$ \frac{d}{dx}[(1-x^2)\frac{d}{dx}U_n(x)] + n(n+1)U_n(x) = 0. $$

They satisfy the following recurrence relation

$$ U_{i+1}(x) = \frac{2i+1}{i+1} x U_i(x) - \frac{i}{i+1} U_{i-1}(x), \quad i = 1,2, \ldots, $$

In the scenario where $U_0(x) = 1$ and $U_1(x) = x$, we can create the translated Legendre polynomials spanning the range $[0,T]$ by applying the linear transformation $x = \frac{2}{T}t - 1$. In this context, $L_m(t)$ represents the translated Legendre polynomials of degree m and they can be derived using the subsequent recurrence relation.

$$ L_{m+1}(t) = \frac{2m+1}{m+1}\left(\frac{2}{T}t - 1\right) L_m(t) - \frac{m}{m+1} L_{m-1}(t), \quad m = 1,2, \ldots, $$

where $L_0(t) = 1$ and $L_1(t) = \frac{2}{T}t - 1$. For any polynomial $L_m(t)$, with $m \in \{0,1, \ldots\}$, its orthogonality in relation to the weight function $h(t) = 1$ is described as

$$ \int_0^T L_m(t) L_n(t) dt = \begin{cases} \frac{T}{2n+1} & m = n, \\ 0 & m \neq n. \end{cases} \tag{8}$$

The shifted Legendre polynomial of degree n, denoted as $L_n(t)$, over the interval $[0,T]$ can be characterized by its analytical representation given by

$$ L_n(t) = \sum_{k=0}^n (-1)^{n+k} \frac{(n+k)!}{(n-k)!(k!)^2} \left(\frac{t}{T}\right)^k. \tag{9}$$

Next, we move forward to introduce the concept of shifted Legendre-Gauss nodes and the associated quadrature weights.

Let t_k, $k = 1,2, \ldots, N-1$, represent the conventional Legendre-Gauss nodes over the interval $(-1,1)$, which are identified as the roots of $L_{N-1}(t)$. One pivotal characteristic of the LG points can be described as

$$ \int_{-1}^1 f(t) dt \simeq \sum_{k=1}^{N-1} c_k f(t_k). \tag{10}$$

This definite integral is accurate for polynomials with a maximum degree of $2N-3$, and the weights for numerical integration can be computed as (Canuto and Hussaini, 2012)

$$ c_k = \frac{2}{(1-t_k^2)[U'_{N-1}(t_k)]^2}, \quad k = 1, \ldots, N-1. $$

Let the Legendre-Gauss nodes, transformed to the interval $[0,T]$, be represented by η_k for $k = 1, \ldots, N-1$. One can derive that $\eta_k = \frac{T}{2}(t_k + 1)$. The associated quadrature weights can be expressed as $\widetilde{w}_k = \frac{T}{2} c_k$. Introducing two distinct points, $\xi_0 = 0$ and $\xi_N = T$, the Lagrange polynomials can be subsequently defined as

$$ P_i(t) = \prod_{l=0, l\neq i}^N \frac{t-\eta_l}{t_i-\eta_l}, \quad i = 0,1, \ldots, N. \tag{11}$$

Consider the shifted Legendre-Gauss nodes over the interval $[a,b]$ represented by ξ_r for $r = 1, \ldots, N-1$. We have $\xi_r = \frac{b-a}{2} t_k + \frac{b+a}{2}$.

Accordingly, the corresponding Lagrange polynomials are defined over the interval $[a, b]$ as

$$P_i(x) = \prod_{l=0, l\neq i}^{N} \frac{x-\xi_l}{x_i-\xi_l}, \quad i = 0,1, \dots, N, \qquad (12)$$

where, $\xi_0 = a$ and $\xi_N = b$.

Design of neural network model

The exploration of mathematical research in biological nervous systems has paved the way for the development of artificial neural networks (ANNs) (Haykin, 1998). Figure 1 shows a simplified model of the neuron as an inspiration for mathematical model. Inspired by the intricate workings of the human brain, ANNs aim to create machines capable of learning and adapting. This research has sparked fascinating advancements in the field of machine learning, where algorithms and models strive to replicate the cognitive processes of human intelligence. By delving into the complexities of biological nervous systems, researchers have unlocked new possibilities for designing intelligent machines that can understand, reason, and learn from data. These artificial machines, mimicking biological neurons, are termed as nodes or artificial neurons. Every artificial neuron has multiple inputs but only one output, linking it to numerous other synthetic neurons. ANNs are widely utilized in the realm of approximation theory. Drawing on the Kolmogorov existence theorem, a suitable neural network can be utilized to approximate any continuous function of n variables (Chen et al., 1995). ANNs have emerged as a powerful instrument in both mathematics and engineering, offering the potential to approximate solutions to a wide range of challenges. Neurons are commonly arranged in several layers. Each neuron produces its output by determining the weighted sum of its input values, which also includes a bias. This particular sum, often termed the net input, undergoes a transformation via an activation function to yield the final output (as shown in Fig. 1). There can be none or multiple hidden layers sandwiched between the input and output layers. If we exclude these hidden layers, we obtain a distinct category of neural networks known as FLNN. Subsequently, we focus on using an FLNN to solve (1)-(3).

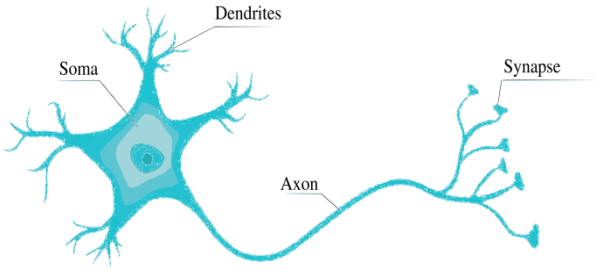
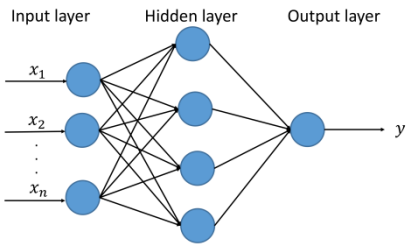


Fig. 1. Model of the biological neuron.

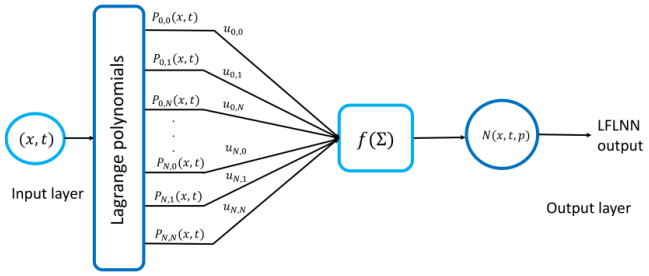
Architecture of Lagrange functional link-based neural network

The FLNNs, often referred to as single-layered neural networks, function without hidden layers. In such configurations, the input data undergoes nonlinear functional expansion for enhancement. This strategy leads to a decrease in computational complexity and offers better approximation capabilities relative to back-propagation techniques (Ghazali et al., 2009). It is important to note that the FLNN models can be trained faster than MLP (Multi-Layer Perceptron) while maintaining computational efficiency.

Moreover, the effectiveness of the LFLNN enhances its attractiveness. With its single-layer design, the model streamlines the learning process, making it faster and more computationally efficient compared to more complex neural network architectures. This makes the LFLNN an attractive choice for applications where computational time is a crucial factor.



(a) Architecture of the neural network



(b) LFLNN design

Fig. 2. Schematic representation of the traditional ANN and the LFLNN.

Formulation of LFLNN for VOFTVDPDEs

The architecture of the LFLNN features a block dedicated to the Lagrange polynomial, connecting a single input node directly to an output node (refer to Fig. 1). The mathematical representation of this relationship can be described using input vectors

$$t = [(x_0, t_0), (x_0, t_1), \dots, (x_0, t_N), \dots, (x_N, t_0), (x_N, t_1), \dots, (x_N, t_N)],$$

and weights

$$w = [u_{0,0}, u_{0,1}, \dots, u_{0,N}, \dots, u_{N,0}, u_{N,1}, \dots, u_{N,N}]$$

as follows

$$N(x,t,w) = \vartheta(q),$$
$$q = \sum_{m=0}^{N} \sum_{n=0}^{N} u_{mn}\, P_m(x) P_n(t).$$

The output function $N(x,t,w)$ can be formulated by introducing the input π to several activation functions. In (Kheyrinataj and Nazemi, 2020), the authors employed the activation function $\vartheta(z) = tanh(z) = \frac{exp(2z)-1}{exp(2z)+1}$ in their FLNN model. This specific activation function has garnered attention in numerous applications (Kheyrinataj & Nazemi, 2019).

The sigmoid function, defined as $\vartheta(z) = \frac{1}{1+exp(-z)}$, is also a frequently utilized activation function. However, for our current problem, these functions might not be optimal due to the computational burden introduced by the exponential terms. Through numerical demonstrations, we will illustrate that the linear activation function $\vartheta(z) = z$ serves as an effective solution for the problem defined by (1)-(3).

Let $u_P(x,t,w)$ represent the neural-based solution to the equations (1)-(3). Hence, it can be expressed as

$$u_P(x,t,w) = \psi(t,x) + t(x-a)(x-b)N(x,t,w), \quad (13)$$

The initial conditions are met by incorporating the first term. The second term represents a single-layer LFLNN with tunable parameters. Accordingly, the solutions from the neural network meet the criteria in Eq. (1), allowing us to express it as

$$\frac{\partial^{q(t)} u_P(x,t,w)}{\partial t^{q(t)}} - \eta \frac{\partial^2 u_P(x,t,w)}{\partial x^2} = f\big(t, u_P(x,t,w), u_P\big(x, t-s(t)\big), w\big), \quad a < x < b, \quad 0 < t \le T. \quad (14)$$

Based on the definition of the variable-order fractional derivative, we obtain

$$\frac{\partial^{q(t)} u_P(x,t,w)}{\partial t^{q(t)}} = \frac{1}{\Gamma(1-q(t))} \int_0^t (t-r)^{-q(t)} \frac{\partial u_P(x,r,w)}{\partial r} dr, \quad 0 < q(t) < 1, \quad (15)$$

We also notice that

$$u_P(x, t-s(t), w) = \begin{cases} \psi(x,t), & x \in (a,b), \quad t < s(t), \\ u_P(x, t-s(t), w), & x \in (a,b), s(t) \le t < T. \end{cases} \quad (16)$$

To solve Eq. (14), we initially define the subsequent squared error function

$$E(x,t,w) = \left[\frac{\partial^{q(t)} u_P(x,t,w)}{\partial t^{q(t)}} - \eta \frac{\partial^2 u_P(x,t,w)}{\partial x^2} - f(t, u_P(x,t,w), u_P(x, t-s(t)), w)\right]^2. \quad (17)$$

To minimize Eq. (17), we present an unconstrained optimization approach using a uniform discretization as described below

$$\min v(w) = \frac{1}{2} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(\frac{\partial^{q(t)} u_P(x_m,t_n,w)}{\partial t^{q(t)}} - \eta \frac{\partial^2 u_P(x_m,t_n,w)}{\partial x^2} - f(t, u_P(x_m,t_n,w), u_P(x_m, t_n - s(t_n)), w)\right)^2, \quad (19)$$

where

$$x_m, \ m = 1,2,\dots,M, \quad \text{and} \quad t_n, n = 1,2,\dots,N, \quad (20)$$

are shifted Legendre-Gauss nodes. For approximating the variable-order fractional partial derivative at collocation points, we utilize the Gaussian quadrature technique as described

$$\frac{\partial^{q(t)} x_G(x_m,t_n,w)}{\partial t^{q(t)}} = \frac{1}{\Gamma(1-q(t_n))} \int_0^{t_n} (t-r)^{-q(t_n)} \frac{\partial x_G(x_m,r,w)}{\partial r} dr, \quad (20)$$

Through this approach, the initial problem (1)-(3) is transformed into an unconstrained optimization problem. This can be solved using established mathematical optimization methods or heuristic strategies like particle swarm optimization, and genetic algorithms (Dabiri et al., 2018). In our study, we employ the modified Newton method to adjust the weights of the LFLNN during the learning phase.

Remark 3.1 When utilizing the linear activation function $\vartheta(z) = z$ in conjunction with a

polynomial initial condition $\phi(t)$, we can directly determine the variable-order fractional derivative from Eq. (7), eliminating the need for approximating the variable-order derivative.

Training process and convergence analysis

The initial step in neural network training involves establishing a starting weight vector. Subsequently, a series of weight vectors is produced. The training process is concluded upon the fulfillment of a predetermined condition, often referred to as the termination criterion. For our learning and weight updating processes, we employ the backpropagation algorithm. Though various iterative approaches such as gradient descent, Newton's method, and the Conjugate gradient are available, we have opted for the modified Newton-Raphson technique to train the neural network (Susanto and Karjanto, 2009). It's pertinent to note that the applicability of the modified Newton-Raphson method is due to the continuous differentiability of the error function. In order to solve the optimization issue presented in (18), we let $B(\delta) = \nabla v(\delta) = 0$. Given $u(\delta) = J_B^{-1}(\delta)B(\delta)$, with J symbolizing the Hessian matrix associated with the cost function $v(\delta)$, the weights can be obtained as follows

$$\delta_{j+1} = \delta_j - J_u^{-1}(\delta_j)u(\delta_j), \quad (21)$$

where, j represents the iteration step employed to update the weights. Subsequently, we will explain the learning process in the following steps.

• Start by setting the initial values for the weights and the input vector $t = [t_0, t_1, \dots, t_m]$. Use $\epsilon > 0$ as the threshold for error tolerance.

• Calculate the output of the LFLNN model $N(x,t,w)$ and determine the weights through a backpropagation method.

$$\delta_{j+1} = \delta_j - J_u^{-1}(\delta_j)u(\delta_j),$$

• If $|\delta_{j+1} - \delta_j| \leq \epsilon$,, proceed to the subsequent step. If not, evaluate the error function and initiate a new training iteration.

• After completing the desired learning process and achieving the intended results, the final network parameters can be saved.

The flowchart for the learning process is depicted in Fig. 3. We show that this learning method reaches the best solution for the given unconstrained optimization problem as expressed by (18).

Theorem 1. Given a sequence $\{\delta_j, j = 1,2, \dots\}$ as described by Eq. (21), and a continuously differentiable function $u: \mathbb{R}^n \to \mathbb{R}^n$ in the vicinity of the optimal solution δ^* where $\nabla v(\delta^*) = 0$, it follows that $\lim_{j\to\infty}\delta_j = \delta^*$.

Proof. See (Golpour Lasaki, Ebrahimi, & Ilie, 2023).

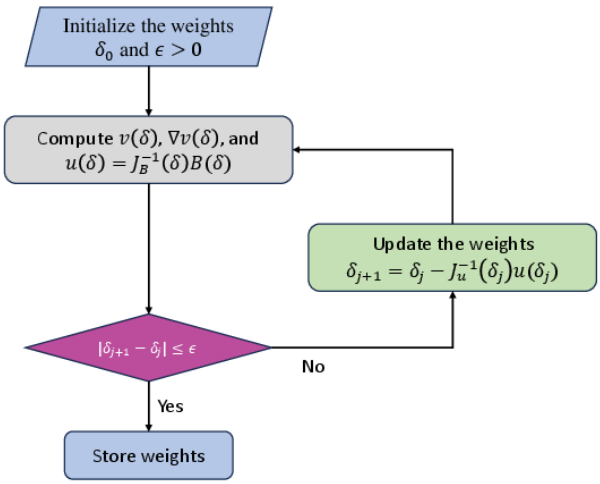


Fig. 3. Diagram illustrating the LFLNN structure's learning algorithm.

Theorem 2. We assume that $u_p(x,t,w)$ is an approximate solution to the equations (1)-(3), and for the derivative operators, we have

$$\| u_{xx}(x,t) - v_{xx}(x,t) \| \leq \lambda_x \| u(x,t) - v(x,t) \|, \quad (22)$$

$$\| u_t(x,t) - v_t(x,t) \| \leq \lambda_t \| u(x,t) - v(x,t) \|, \quad (23)$$

where λ_x and λ_t are positive real numbers. For $\lambda_t I - |\eta|\lambda_x - 2L > 1$, we have $\| E \| \leq K \| r(x,t) \|$, where $K = \frac{1}{\lambda_t I - |\eta|\lambda_x - 2L}$, and as a result, we obtain

$$\lim_{N\to\infty} u_p(x,t,w) = u(x,t), \quad (24)$$

where

$$u_p(x,t,w) = \psi(x,t) + t(x-a)(x-b)N(x,t,w), \quad (25)$$

$$\| E \| = \max_{(x,t)\in(a,b)\times[-s,T]} \{\| e(x,t) \|, \| e(x,t-s(t)) \|\}. \quad (26)$$

Proof. Considering the exact solution $u(x,t)$ and the approximate solution $u_p(x,t,w)$, substituting into the differential equation (1), we have

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \eta\frac{\partial^2 u(x,t)}{\partial t^2} - f(t,u(x,t),u(x,t-s(t))) = 0, \quad (27)$$

$$\frac{\partial^{q(t)}u_p(x,t,w)}{\partial t^{q(t)}} - \eta\frac{\partial^2 u_p(x,t,w)}{\partial t^2} - f(t,u_p(x,t,w),u(x,t-s(t),w)) = r(x,t). \quad (28)$$

By subtracting equation (28) from (27), we get

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^{q(t)}u_p(x,t,w)}{\partial t^{q(t)}} + \eta(\frac{\partial^2 u(x,t)}{\partial t^2} - \frac{\partial^2 u_p(x,t,w)}{\partial t^2}) + f(t,u(x,t),u(x,t-s(t))) - f(t,u_p(x,t,w),u_p(x,t-s(t),w)) = -r(x,t). \quad (29)$$

Now, by considering the Lipschitz condition for the function f, we have

$$\| f(t,u(x,t),u(x,t-s(t))) - f(t,u_p(x,t,w),u_p(x,t-s(t),w)) \| \leq L_1(x,t) \| u(x,t) - u_p(x,t,w) \| + L_2(x,t) \| u(x,t-s(t)) - u_p(x,t-s(t),w) \| = L_1(x,t) \| e(x,t) \| + L_2(x,t) \| e(x,t-s(t)) \|, \quad (30)$$

where $L_i(x,t), i = 1,2$ are positive functions over the domain of the equation. Assuming $D = (a,b)\times[-s,T]$ and

$$\| E \| = \max_{(x,t)\in D}\{\| e(x,t) \|, \| e(x,t-s(t)) \|\}, \quad (31)$$

$$L = \max_{(x,t)\in D}\{L_1(x,t), L_2(x,t)\}. \quad (32)$$

Equation (30) can be written as

$$||f\left(t,u(x,t),u\left(x,t-s(t)\right)\right) - f\left(t,u_p(x,t,w),u_p(x,t-s(t),w)\right)\| \leq 2L \| E \| \quad (33)$$

Also, considering the assumption of the theorem, we have

$$\| \frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^{q(t)}u_p(x,t,w)}{\partial t^{q(t)}} \| \leq \frac{1}{\Gamma(1-q(t))}\int_0^t |t-s|^{q(t)} \| \frac{\partial u(x,s)}{\partial s} - \frac{\partial u_p(x,s,w)}{\partial s} \| ds \leq \frac{1}{\Gamma(1-q(t))}\int_0^t |t-s|^{q(t)}\lambda_t \| \frac{\partial u(x,s)}{\partial s} - \frac{\partial u_p(x,s,w)}{\partial s} \| ds \leq \frac{\lambda_t\|E\|}{\Gamma(1-q(t))}\int_0^t |t-s|^{q(t)}ds \leq \lambda_t I \| E \|, \quad (34)$$

where

$$I = \max_{t\in[-s,T]}\{\int_0^t |t-s|^{q(t)}ds\}.$$

Now, considering equations (27) to (34), we have

$$\| \frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^{q(t)}u_p(x,t,w)}{\partial t^{q(t)}} \| \leq |\eta| \| \frac{\partial^2 u(x,t)}{\partial t^2} - \frac{\partial^2 u_p(x,t,w)}{\partial t^2} + ||f(t,u(x,t),u(x,t-s(t))) - f(t,u_p(x,t,w),u_p(x,t-s(t),w)) \| + \| r(x,t) \|. \quad (35)$$

Then

$$\lambda_t I \| E \| \leq |\eta|\lambda_x \| E \| + 2L \| E \| + \| r(x,t) \|,$$
$$\Rightarrow (\lambda_t I - |\eta|\lambda_x - 2L) \| E \| \leq \| r(x,t) \|. \quad (36)$$

Considering that $\lambda_t I - |\eta|\lambda_x - 2L > 0$, we have

$$\| E \| \leq K \| r(x,t) \|,$$

where $K = \frac{1}{\lambda_t I - |\eta|\lambda_x - 2L}$. Now, if $N \to \infty$, we have $\| r(x,t) \| \to 0$, and consequently

$$\| E \| \to 0 \Rightarrow \lim_{N\to\infty} u_p(x,t,w) = u(x,t).$$

NUMERICAL EXAMPLES

In this section, we assess the effectiveness of the introduced neural network model using multiple numerical examples. The simulations were executed on an X64-based PC equipped with an Intel(R) Core i7 CPU, clocked at 3.10 GHz and 4.0GB RAM, using MAPLE 18 with precision up to 25 decimal digits. For the learning phase, we set a tolerance of $\epsilon = 10^{-4}$. The numerical method's performance is evaluated by computing the absolute error as follows

$$\eta(x,t) = |u(x,t) - x_G(x,t,w)|.$$

Example 1. For the first example, we examine the following VOFTVDPDEs

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^2 u(x,t)}{\partial x^2} = -u\big(x,t-s(t)\big) + \frac{\Gamma(3)(2x-x^2)t^{2-q(t)}}{\Gamma(3-q(t))} + 2t^2 + x(2-x)(t-s(t))^2, \quad 0<x<2, 0<t\leq 1, \quad (37)$$

$$u(x,t) = t^2(2x-x^2), \ x\in(0,2), \ t\in[-s,0], \quad (38)$$

$$u(0,t) = 0, \ u(2,t) = 0, \ t\in(0,1), \quad (39)$$

where $s = \max_{0\leq t\leq T}\{s(t)\}$. The exact solution for this problem is $u(x,t) = t^2(2x-x^2)$. The solution can be approximate as

$$u_p(x,t,w) = t^2(2x-x^2) + tx(x-2)N(x,t,w).$$

This problem has been solved for for $s(t) = 1$ and constant fractioanl-order $q(t) = q$ in

(Dehestani et al., 2019). In Fig. 4, the numerical solution using the suggested method for $q(t) = 1 - \frac{exp(t)}{4}$ and $s(t) = 0.05exp(t)$ with $N = 3$, $M = 10$ are plotted. Figure 5 shows the absolute error function for $q(t) = 0.5$ and $s(t) = 1$ with $N = 3, m = 10$. Table 1 shows the comparison of the results for the proposed methods and the method described in (Dehestani et al., 2019). The numerical results indicate that the method we proposed exhibits superior performance compared to the approach detailed in (Dehestani et al., 2019).

Table 1: A comparative analysis of the absolute error for the suggested method and method (Dehestani et al., 2019). for $q(t) = 0.5$ and $s(t) = 1$ in Example 1.

(x, t)	***Suggested method***	***Method* (Dehestani et al., 2019).**
(0, 0)	0	4.31×10^{-17}
(0.2, 0.1)	0	6.12×10^{-16}
(0.4, 0.2)	0	1.36×10^{-15}
(0.6, 0.3)	0	2.35×10^{-15}
(0.8, 0.4)	0	3.62×10^{-15}
(1, 0.5)	0	5.21×10^{-15}
(1.2, 0.6)	5.551115×10^{-17}	7.18×10^{-15}
(1.4, 0.7)	0	9.57×10^{-15}
(1.6, 0.8)	1.110223×10^{-16}	1.24×10^{-14}
(1.8, 0.9)	5.551115×10^{-17}	1.58×10^{-14}
(2, 1)	7.731641×10^{-24}	1.97×10^{-14}

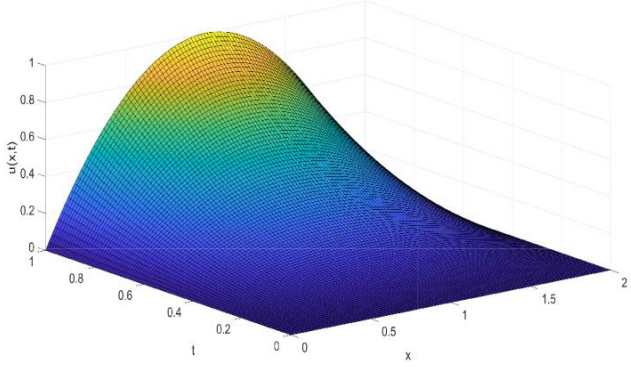


Fig. 4. Approximate solution for $q(t) = 1 - \frac{exp(t)}{4}$ and $s(t) = 0.05exp(t)$ with $N = 3$, $M = 10$ in Example 1.

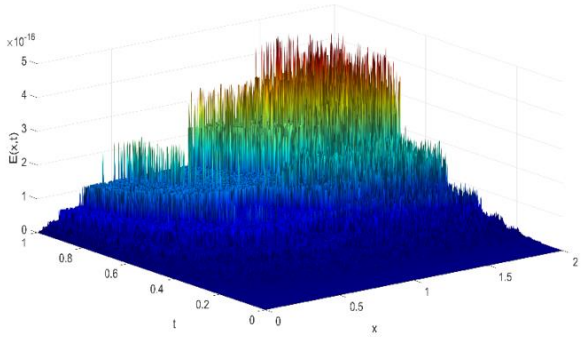


Fig. 5. Absolute error of the proposed method for $q(t) = 0.5$ and $s(t) = 1$ with $N = 3$, $M = 10$ in Example 1.

Example 2. We consider the following VOFTVDPDEs

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^2 u(x,t)}{\partial x^2} = u(x, t - s(t)) - 2t^2 + \frac{\Gamma(3)t^{2-q(t)}}{\Gamma(3-q(t))}x^2 - x^2(t - s(t))^2, \quad (40)$$

$$0 < x < 1, \quad 0 < t \leq 2,$$

$$u(x,t) = x^2t^2, \ x \in (0,1), \ t \in [-1,0], \quad (41)$$

$$u(0,t) = 0, u(1,t) = t^2, \quad t \in (0,2), \quad (42)$$

The exact solution for this problem is $u(x, t) = x^2t^2$. The solution derived from the neural networks approach can be expressed as $u_p(x, t, w) = x^2t^2 + tx(x - 1)N(x, t, w)$.

In Fig. 6, the numerical results using the suggested method for $q(t) = 1 - \frac{exp(t)}{4}$ and $s(t) = 0.05exp(t)$ with $N = 3$, $M = 10$ are depicted. Figure 7 illustrates the absolute error function for the $q(t) = 1 - \frac{exp(t)}{4}$ and $s(t) = 0.05exp(t)$ with $N = 3$, $M = 10$. Table 2 reported the value of the $\min \nu(w)$ for different values of $q(t)$. As can be seen from the results, the $\min \nu(w)$ is close to zero with respect to different values of $q(t)$ which demonstrates the effectiveness of the numerical method.

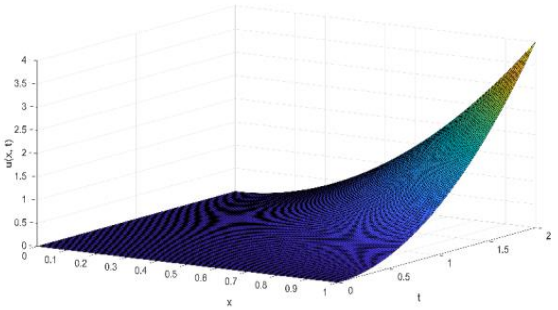


Fig. 6. Approximate solution for $q(t) = 1 - \frac{exp(t)}{4}$ and $s(t) = 0.05exp(t)$ with $N = 3$, $M = 10$ in Example 2.

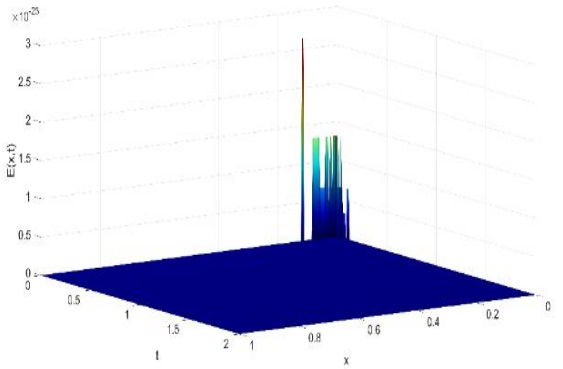


Fig. 7: Absolute error of the $q(t) = 0.5$ and $s(t) = 0.05exp(t)$ with $N = 3$, $M = 10$ in Example 4.2.

Table 2: Value of min ν(w) for different values of q(t) in Example 2.

$\boldsymbol{q(t)}$	$\boldsymbol{\min \nu(w)}$
$1 - \frac{exp(t)}{4}$	8.317647×10^{-38}
$1 - \frac{cos(t)^2}{4}$	5.704133×10^{-34}
$1 - 0.25t$	6.134233×10^{-38}

Example 3. As the third example, consider the following VOFTVDPDE

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^2 u(x,t)}{\partial x^2} = u\big(x, t - s(t)\big) + \frac{2xt^{1-q(t)}}{\Gamma(2-q(t))} + \frac{\Gamma(3)t^{2-q(t)}}{\Gamma(3-q(t))} - (x + t - s(t))^2 - 2, \quad 0 < x < 2,\ 0 < t \leq 1, \tag{43}$$

$$u(x,t) = (x+t)^2,\ x \in (0,2),\ t \in [-1,0], \tag{44}$$

$$u(0,t) = t^2,\ u(2,t) = (2+t)^2,\ t \in (0,1), \tag{45}$$

The exact solution for this problem is represented by the equation $u(x,t) = (x+t)^2$. In terms of a neural-based approach, the solution can be expressed as $u_p(x,t,w) = (x+t)^2 + tx(x-2)N(x,t,w)$. where $q(t) = 1 - \frac{cos(t)^2}{2}$ and $s(t) = cos(t)exp(t)$. In Fig. 8, the numerical solution using the suggested method for $q(t) = 1 - \frac{cos(t)^2}{2}$ and $s(t) = cos(t)exp(t)$ with $N = 4$, $M = 10$ are shown. Figure 9 shows the absolute error function for the $q(t) = 1 - \frac{cos(t)^2}{2}$ and $s(t) = cos(t)exp(t)$ with $N = 4$, $M = 10$. Table 4.3 presents a comparison based on different values of N for absolute error of the suggested method. The results indicate the absolute error decreases with increasing the value of the N. The minimum values of the cost function $\nu(w)$ for different values of N have been reported in Table 4. The results also show that the minimum of $\nu(w)$ decreases as the value of N increases.

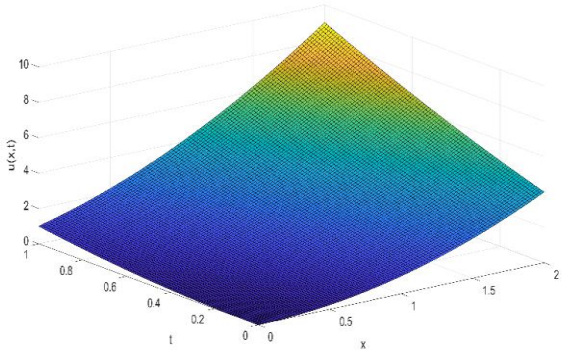


Fig. 8. Approximate solution for $q(t) = 1 - \frac{cos(t)^2}{2}$ and $s(t) = cos(t)exp(t)$ with $N = 4$, $M = 10$ in Example 3.

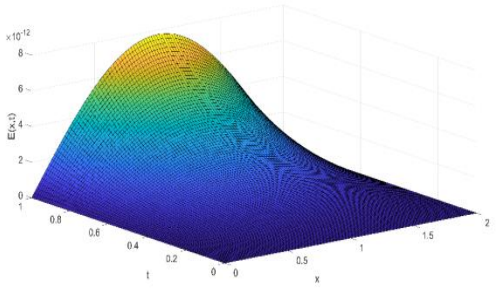


Fig. 9. Absolute error of the $q(t) = 1 - \frac{cos(t)^2}{2}$ and $s(t) = cos(t)exp(t)$ with $N = 4$, $M = 10$ in Example 3.

Table 3: Absolute errors for different values of N in Example 1.

Points	$N = 4$	$N = 5$	$N = 6$
(0, 0)	0	0	0
(0.2, 0.1)	4.446443 10^{-14}	0	0
(0.4, 0.2)	1.845191 10^{-13}	0	0
(0.6, 0.3)	4.543033 10^{-13}	1.110223 10^{-16}	1.110223 10^{-16}
(0.8, 0.4)	8.772982 10^{-13}	0	0
(1, 0.5)	1.433076 10^{-12}	0	0
(1.2, 0.6)	2.023715 10^{-12}	4.440892 10^{-16}	4.440892 10^{-16}
(1.4, 0.7)	2.467360 10^{-12}	8.881784 10^{-16}	8.881784 10^{-16}
(1.6, 0.8)	2.498446 10^{-12}	1.776357 10^{-15}	0
(1.8, 0.9)	1.788791 10^{-12}	8.881784 10^{-16}	0
(2, 1)	1.776357 10^{-15}	0	0

Table 4: Value of $min\ \nu(w)$ for different values of N in Example 3.

N	$\min \nu(w)$
$N = 4$	2.311611×10^{-25}
$N = 5$	5.125486×10^{-37}
$N = 6$	3.916432×10^{-37}

Example 4. Now, we consider the following VOFTVDPDE

$$\frac{\partial^{q(t)}u(x,t)}{\partial t^{q(t)}} - \frac{\partial^2 u(x,t)}{\partial x^2} = u(x, t - s(t)) + \frac{\Gamma(3)t^{2-q(t)}}{\Gamma(3-q(t))}(x+1)^3 - 6t^2(x+1) - (t - s(t))^2(x+1)^3,\ 0 < x < 1,\ 0 < t \le 1, \quad (46)$$

$$u(x,t) = t^2(x+1)^3,\ t \in [-1,0], \quad (47)$$

$$u(0,t) = t^2, \quad u(2,t) = 8t^2,\ t \in (0,1), \quad (48)$$

The exact solution to this problem is given by $u(x,t) = t^2(x+1)^3$. The neural-based solution can be expressed as

$$u_p(x,t,w) = t^2(x+1)^3 + tx(x-1)N(x,t,w).$$

where $q(t) = 1 - 0.5t$ and $s(t) = \frac{|cos(t)|}{2}$. In Fig. 10, the numerical solution using the suggested method with $N = 4$, $M = 10$ are plotted. Figure 11 shows the absolute error function for the $q(t) = 1 - 0.5t$ and $s(t) = \frac{|cos(t)|}{2}$ with $N = 4$, $M = 10$. Table 5 presents the minimum values of $\nu(w)$. It shows that as the value of N increases, the minimum value of $\nu(w)$ decreases.

Table 5: Value of $min\ \nu(w)$ for different values of N in Example 4.

N	$\min \nu(w)$
$N = 2$	2.310241×10^{-35}
$N = 3$	1.667555×10^{-35}
$N = 4$	7.618161×10^{-36}

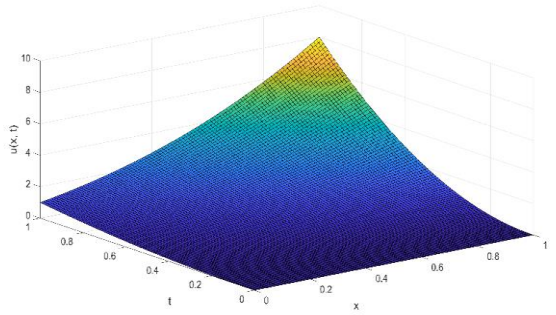


Fig. 10. Approximate solution for $q(t) = 1 - 0.5t$ and $s(t) = cos(t)exp(t)$ with $N = 4$, $M = 10$ in Example 4.

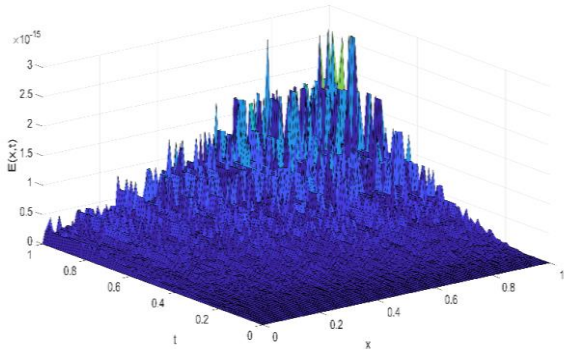


Fig. 11. Absolute error of the $q(t) = 1 - 0.5t$ and $s(t) = cos(t)exp(t)$ with $N = 4$, $M = 10$ in Example 4.

Example 5. As the final example, we consider the following time-delay variable-order fractional partial differential equation

$$\frac{\partial^{\alpha(t)}u(x,t)}{\partial t^{\alpha(t)}} = \frac{\partial^2 u(x,t)}{\partial x^2} + u\big(x, t - \tau(t)\big) + sin(\pi x)\left(\frac{2t^{2-\alpha(t)}}{\Gamma(3-\alpha(t))} - (t-1)^2 + \pi^2 t^2\right),\ 0 < x < 1,\ 0 < t \le 1, \quad (49)$$

$$u(x,t) = t^2 sin(\pi x),\ ,t \in [-1,0], \quad (50)$$

$$u(0,t) = 0,\ u(1,t) = 0,\ t \in (0,1), \quad (51)$$

where $\alpha(t) = 0.5 + 0.5t$ and $\tau(t) = 1$. The exact solution for this problem is $u(x,t) = t^2 sin(\pi x)$.

The approximate solution using the proposed neural network can be written as

$$u_p(x,t,w) = t^2 sin(\pi x) + tx(x-1)N(x,t,w). \quad (52)$$

Figures 12 and 13 show the numerical solution and the absolute error for $\alpha(t) = 0.5 + 0.5t$ and $\tau(t) = 1$ with $N = 3$ and $M = 10$, respectively. Table 6 presents the values of $\Theta(w)$ for various values of N.

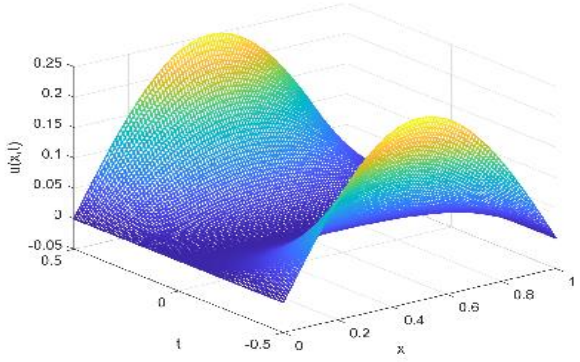


Fig. 12. Numerical solution for $\alpha(t) = 0.5 + 0.5t$ and $\tau = 1$ with $N = 3$ and $M = 10$ in Example 5.

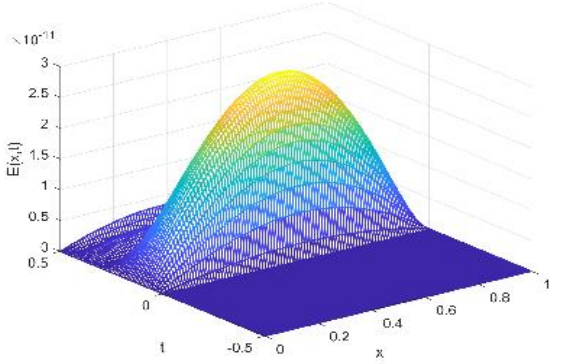


Fig. 13: Absolute error for $\alpha(t) = 0.5 + 0.5t$ and $\tau = 1$ with $N = 3$ and $M = 10$ in Example 5.

Table 6: Value of min $\Theta(w)$ for $M = 10$ and various values of N in Example 4.5.

N	$min\ \Theta(w)$
$N = 3$	5.630073×10^{-17}
$N = 4$	4.553892×10^{-17}
$N = 5$	3.769620×10^{-17}

DISCUSSION OF RESULTS

The findings suggest that the LFLNN is capable of effectively solving VOTVFPDEs. The capability to solve VOTVFPDEs provides solutions to an array of challenges, especially those associated with constant derivative order and time lags in dynamic models. Such challenges are ubiquitous in engineering and science. To enhance the outcome quality, we incorporated Lagrange basis functions and LG points during discretization. Throughout the learning phase, preference was given to the modified Newton-Raphson technique over the typically employed gradient descent method, which has shown to be proficient for VOTVFPDEs.

A notable feature of the LFLNN is its ability to yield good results with just a few basis functions. Furthermore, the integration of Lagrange polynomials as foundational functions in the LFLNN framework eliminates the need for hidden layers. Thus, in comparison to MLP neural networks, the LFLNN executes outputs more rapidly, leading to a decrease in computational burden.

Prospective explorations can focus on employing an FLNN to solve fractional optimal control problems with time-varying delay. Furthermore, cutting-edge metaheuristic algorithms like particle swarm optimization, ant colony optimization whale, and lion optimization algorithms can be used to solve the optimization problem.

CONCLUSION

In the current study, we introduced an advanced LFLNN framework complemented by a novel optimization approach specifically formulated for VOTVFPDEs. Using the Newton-Raphson method as a departure from the common gradient descent paradigm, our methodology exhibited a great accuracy in absolute error metrics. A key feature of our work is the attainment of high precision using a minimal set of Lagrange polynomials within the LFLNN structure, which is pivotal, considering the risk of overfitting with too many polynomials. Given the accuracy, our proposed methodology demonstrates itself as a promising framework for solving challenges posed by VOTVFPDEs. While this method is designed for VOTVFDDEs, it has great potential for use in many areas of engineering and science.

FUNDING

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

CONFLICT OF INTEREST

This article does not contain any studies with human participants or animals performed by any of the authors.

ETHICAL APPROVAL

The authors have no conflict of interest to declare.

REFERENCES

Akgül, A., Inc, M., & Baleanu, D. (2017). On solutions of variable-order fractional differential equations. *International Journal of Optimization and Control Theories & Applications*, *7*(1), 12-26. https://doi.org/10.11121/ijocta.01.2017.00368

Al-Janabi, S., & Alkaim, A. (2020). A nifty collaborative analysis to predicting a novel tool (drflls) for missing values estimation. *Soft Computing*, 24 (1), 38-46. https://doi.org/10.1007/s00500-019-03972-x

Al-Janabi, S., Alkaim, A., & Adil, Z. (2020). An innovative synthesis of deep learning techniques (DCapsNet & DCOM) for generation electrical renewable energy from wind energy. *Methodologies and Application*, 24 (1), 16-24. https://doi.org/10.1007/s00500-020-04905-9

Al-Janabi, S., Mohammad, M., & Al-Sultan, A. (2020). A new method for prediction of air pollution based on intelligent computation. *Soft Computing*, 24 (1), 62-76. https://doi.org/10.1007/s00500-019-04495-1

Bijiga, L. K., & Ibrahim, W. (2021). Neural network method for solving time-fractional telegraph equation. *Mathematical Problems in Engineering*, 12(2), 78-87. https://doi.org/10.1155/2021/7167801

Canuto, C., Hussaini, M. Y., Quarteroni, A., & Thomas, A. Jr. (2012). Spectral methods in fluid dynamics. *Springer Science & Business Media*. ISBN: 978-3-642-84108-8.

Chávez-Vázquez, S., Gómez-Aguilar, J. F., Lavín-Delgado, J. E., Escobar-Jiménez, R. F., & Olivares-Peregrino, V. H. (2022). Applications of Fractional Operators in Robotics: A Review. *Journal of Intelligent & Robotic Systems*, 104(63). https://doi.org/10.1007/s10846-022-01597-1

Choudhary, R., Singh, S., & Kumar, D. (2022). A second-order numerical scheme for the time-fractional partial differential equations with a time delay. *Computational and Applied Mathematics*, 88(1), 114-224. https://doi.org/10.1007/s40314-022-01810-9

Dabiri, A., Moghaddam, B. P., & Machado, J. A. T. (2018). Optimal variable-order fractional PID controllers for dynamical systems. *Journal of Computational and Applied Mathematics*, 339, 40-48. https://doi.org/10.1016/j.cam.2018.02.029

Dehestani, H., Ordokhani, Y., & Razzaghi, M. (2019). On the applicability of Genocchi wavelet method for different kinds of fractional-order differential equations with delay. *Numerical Linear Algebra with Applications*, 17 (1), 18-29. https://doi.org/10.1002/nla.2259

Diethelm, K., Kiryakova, V., Luchko, Y., Machado, J. A. T., & Tarasov, V. E. (2022). Trends, directions for further research, and some open problems of fractional calculus. *Nonlinear Dynamics*, 107(1), 3245-3270.

Ghazali, R., Hussain, A. J., Al-Jumeily, D., & Lisboa, P. (2009). Time series prediction using dynamic ridge polynomial neural networks. In 2009 Second International Conference on Developments in E-systems Engineering (pp. 354–363). IEEE. https://doi.org/10.1109/DeSE.2009.35

Golpour Lasaki, F., Ebrahimi, H., & Ilie, M. (2023). A novel Lagrange functional link neural network for solving variable-order fractional time-varying delay differential equations: a comparison with multilayer perceptron neural networks. *Soft Computing*, 27 (1), 12595-12608. https://doi.org/10.1007/s00500-023-08494-1

Hattaf, K. (2022). On the stability and numerical scheme of fractional differential equations with application to biology. *Computations*, 10(6). 108-126. https://doi.org/10.3390/computation10060097

Haykin, S. (1998). Neural Networks: A Comprehensive Foundation (2nd ed.). Pearson. ISBN-13: 978-0132733502.

Chen, T., Chen, H., & Liu, R. W. (1995). Approximation capability in multilayer feedforward networks and related problems. *IEEE Transactions on Neural Networks*, 6(1), 25-30. https://doi.org/10.1109/72.363453

Heydari, M. H., & Avazzadeh, Z. (2018). Legendre wavelets optimization method for variable-order fractional Poisson equation. Chaos, *Solitons & Fractals*, 112 (1), 180-190. https://doi.org/10.1016/j.chaos.2018.04.028

Hosseinpour, S., Nazemi, A., & Tohidi, E. (2018). A new approach for solving a class of delay fractional partial differential equations. Mediterranean. *Journal of Mathematics*, 15(1), 1-20. https://doi.org/10.1007/s00009-018-1264-z

Kadhuim, Z. A., & Al-Janabi, S. (2023). Codon-mrna prediction using deep optimal neurocomputing technique (dlstm-dsn-woa) and multivariate analysis. *Results in Engineering*, 17(1), 118-132. https://doi.org/10.1016/j.rineng.2022.100847

Kheyrinataj, F., & Nazemi, A. (2019). Fractional power series neural network for solving delay fractional optimal control problems. *Connection Science*, 32(1), 1-28. https://doi.org/10.1080/09540091.2019.1605498

Kheyrinataj, F., & Nazemi, A. (2020). Fractional Chebyshev functional link neural network-optimization method for solving delay fractional optimal control problems with Atangana- Baleanu derivative. *Optimal Control Applications and Methods*, 27(1), 246-261. https://doi.org/10.1002/oca.2572

Kumar, P., & Erturk, V. S. (2022). The analysis of a time delay fractional COVID-19 model via Caputo type fractional derivative. *Mathematical Methods in the Applied Sciences*, 46(7), 23-213. https://doi.org/10.1002/mma.6935

Moghaddam, B. P. and Machado, J.T. (2017). A stable three-level explicit spline finite difference scheme for a class of nonlinear time variable order fractional partial differential equations. *Computer & Mathematics with Applications*, 73(6), 1262-1269. https://doi.org/10.1016/j.camwa.2016.07.010

Mohammed, G. S., & Al-Janabi, S. (2020). An innovative synthesis of optimization techniques (fdire-gsk) for generation electrical renewable energy from natural resources. *Results in Engineering*, 16(1), 118-126. https://doi.org/10.1016/j.rineng.2022.100637

Pao, Y.-H. (1989). Adaptive Pattern Recognition and Neural Networks. Addison-Wesley Longman Publishing Co., Inc. ISBN-13: 978-0201125849.

Patnaik, S., Hollkamp, J. P., & Semperlotti, F. (2020). Applications of variable-order fractional operators: a review. Proceedings of the Royal Society A: Mathematical, *Physical and Engineering Science*, 476(1), 226-238. https://doi.org/10.1098/rspa.2019.0498

Patra, J. C., Chin, W. C., Meher, P. K., & Chakraborty, G. (2008). Legendre-flann-based nonlinear channel equalization in wireless communication system. In 2008 IEEE International Conference on Systems, Man and Cybernetics (pp. 1826–1831). https://doi.org/10.1109/icsmc.2008.4811554

Peterson, L. E., & Larin, K. V. (2008). Hermite/laguerre neural networks for classification of artificial fingerprints from optical coherence tomography. In 2008 Seventh International Conference on Machine Learning and Applications (pp. 637–643). https://doi.org/10.1109/ICMLA.2008.36

Sheng, H., Sun, H., Chen, Y. Q., & Qiu, T. Sh. (2011). Synthesis of multifractional Gaussian noises based on variable-order fractional operators. *Signal Processing*, 91(7), 1645–1650. https://doi.org/10.1016/j.sigpro.2011.01.010

Singh, A. K., Mehra, M., & Gulyani, S. (2021). A modified variable-order fractional SIR model to predict the spread of COVID-19 in India. *Mathematical Methods in Applied Science*, 46(7), 240-248. https://doi.org/10.1002/mma.7655

Susanto, H., & Karjanto, N. (2009). Newton's method basin of attraction revisited. *Applied Mathematics and Computation*, 215(1), 1084-1090. https://doi.org/10.1016/j.amc.2009.06.041

Syah, R., Guerrero, J. W. G., Poltarykhin, A. L., Suksatan, W., Aravindhan, S., Bokov, D. O., Abdelbasset, W. K., Al-Janabi, S., Alkaim, A. F., & Tumanov, D. Y. (2022). Developed teamwork optimizer for model parameter estimation of the proton exchange membrane fuel cell. *Energy Reports*, 8(1), 10776–10785. https://doi.org/10.1016/j.egyr.2022.08.177

Tayebi, A., Shekari, Y., & Heydari, M. H. (2017). A meshless method for solving two-dimensional variable-order time fractional advection-diffusion equation. *Journal of Computational Physics*, 340(1), 655-669. https://doi.org/10.1016/j.jcp.2017.03.061

Zou, A.-M., Kumar, K. D., & Hou, Z.-G. (2010). Quaternion-based adaptive output feedback attitude control of spacecraft using chebyshev neural networks. *IEEE Transactions on Neural Networks*, 21(7), 1457-1471. https://doi.org/10.1109/TNN.2010.2050333

Zúniga-Aguilar, C., Gómez-Aguilar, J., Escobar Jiménez, R., & Romero Ugalde, H. (2019). A novel method to solve variable-order fractional delay differential equations based in lagrange interpolations. *Chaos*, 29(6), 146-168. https://doi.org/10.1016/j.chaos.2019.06.009