Networks and Heterogeneous Media

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

NHM, 18(1): 494–512.
DOI: 10.3934/nhm.2023020
Received: 15 November 2022
Revised: 25 December 2022
Accepted: 26 December 2022
Published: 16 January 2023

Research article

SLeNN-ELM: A shifted Legendre neural network method for fractional delay differential equations based on extreme learning machine

Yinlin Ye, Yajing Li *, Hongtao Fan, Xinyi Liu and Hongbing Zhang

College of Science, Northwest A&F University, Yangling, Shaanxi 712100, China

* **Correspondence:** Email: hliyajing@163.com.

Abstract: In this paper, we introduce a shifted Legendre neural network method based on an extreme learning machine algorithm (SLeNN-ELM) to solve fractional differential equations with constant and proportional delays. Based on the properties of Caputo fractional derivatives and shifted Legendre polynomials, the fractional derivatives of SLeNN can be represented analytically without other numerical techniques. SLeNN, in terms of neural network architecture, uses a function expansion block to replace the hidden layer, and thus improving the computational efficiency by reducing parameters. In terms of solving technology of neural networks, the extreme learning machine algorithm is used to replace the traditional gradient-based training algorithm. It dramatically improves our solution efficiency. In addition, the proposed method does not require parameter initialization randomly, making the neural network solution stable. Finally, three examples with constant delays and three examples with proportional delays are given, and the effectiveness and superiority of the proposed method are verified by comparison with other numerical methods.

Keywords: Fractional delay differential equation; Constant delay and proportional delay; Shifted Legendre neural network; Extreme learning machine algorithm

1. Introduction

In the last decades, it has been noticed that many phenomena usually modeled by ordinary differential equations (ODEs) can be better modeled by delay differential equations (DDEs) [1]. Unlike ODEs, the solution of DDEs requires not only information about the current state, but also some information about the previous state. Currently, DDEs play an important role in many practical areas, especially in population dynamics, infectious diseases and chemical kinetics, see [2–4].

In recent years, fractional differential equations have superior modeling capabilities in various scientific and engineering fields, and fractional delay differential equations (FDDEs) have been widely concerned by researchers [5, 6]. Numerous studies have shown that FDDEs are applicable to

different problems in many fields such as population dynamics, economy, control, medicine, electrodynamics and chemistry [7, 8].

Since the analytic solution of most FDDEs cannot be represented explicitly, the intensive focus of the researchers is to achieve its numerical solutions [9]. Over the years, some numerical techniques for solving FDDEs have been developed. These methods include method of steps [10], Adams-Bashforth-Moulton method [11], Runge-Kutta-type method [12], finite difference method [13], Bernoulli wavelets [14], Bernstein operational matrix of differentiation method [15] and modified Chebyshev wavelet method [16]. However, the above techniques do not guarantee that the solutions of differential equations are analytical, and in this respect, neural network methods are superior to classical numerical techniques [17].

Differential equations have been solved by neural network methods for more than 20 years [18]. Here, we briefly review some critical advances in neural network methods for differential equations. Lagaris et al. [19] present a method for solving initial and boundary problems for ODEs, coupled ODE systems, and partial differential equations using artificial neural networks as a trial solution. Jafarian et al. [20] use the generalized sigmoid function as the cost function and the three-layer feedforward structure is considered an iterative scheme for solving linear fractional ODEs. Hou et al. [21] study the use of neural networks for various pantograph-type functional differential equation problems with a proportional delay term subject to initial or boundary conditions. Shiri et al. [22] propose a neural network method to solve fractional diffusion equations using an adaptive gradient descent method to minimize the energy function. Recently, Ye et al. [23] introduce a deep neural network approach to solve time-fractional differential equations in the conformable sense. In addition, other advances in fractional-order time-delay neural networks can be found in [24–26].

Compared with classical numerical methods, neural network methods offer the following advantages:

- Neural networks trained can obtain the value of the solution of the differential equation at any position in the definition domain.
- Due to the representability of neural networks, the resulting neural network solutions are analytical in essence. That is, these solutions are available in the form of continuously differentiable functions.
- Neural network methods possess high solution accuracy and good generalization properties.
- A similar procedure is employed for different initial and boundary conditions of the differential equations.
- The computational complexity of neural network methods does not increase rapidly with more sampled training points.

Up to present, efforts to invest in FDDEs remain low compared to other fields [27]. Furthermore, to the best of our knowledge, no scholars have used neural network methods to solve FDDEs. We therefore aim to propose an accurate and efficient numerical method to solve FDDEs. This is the motivation of this paper. As a family of orthogonal functions, the Legendre family behaves very efficiently in numerical calculations [28]. Yang et al. proposed a Legendre neural network for function approximation to avoid the complexity of traditional feedforward neural networks [29]. Both Saadatmandi et al. [30] and Qu et al. [31, 32] studied the methods based on Legendre polynomials for solving fractional differential equations. In addition, Legendre neural network has been successfully

applied in channel equalization [33] and system identification fields [34]. Inspired by these, we propose a shifted Legendre neural network (SLeNN) to solve the following two problems of FDDEs:

1. Constant delay

$$\begin{cases} {}_0^C D_t^\alpha x(t) = h_1(t) + h_2(t)x(t) + h_3(t)x(t-\tau), & t \in (0, T], \\ x(t) = \phi(t), & t \le 0, \\ x^{(p)}(0) = \lambda_p, & p \in \{0, 1, \ldots, n-1\}, \end{cases} \tag{1.1}$$

2. Proportional delay

$$\begin{cases} {}_0^C D_t^\alpha x(t) = g_1(t) + g_2(t)x(t) + g_3(t)x(\tau t), & t \in (0, T], \\ x^{(p)}(0) = \lambda_p, & p \in \{0, 1, \ldots, n-1\}, \end{cases} \tag{1.2}$$

where $\tau \in (0, 1]$, $\alpha \in (n-1, n]$, $n \in \mathbb{N}$, $x^{(p)}$ represents the p-derivative of x and $\lambda_p \in \mathbb{R}$ for $p \in \{0, 1, \ldots, n-1\}$, and $\phi(0) = \lambda_0$.

The proposed SLeNN is a functional link neural network (FLNN) based on shifted Legendre polynomials, which enhances the input by shifted Legendre polynomials and skips the hidden layer to improve computing efficiency. The output of SLeNN is used as the trial solution for FDDEs. Unlike expensive gradient-based optimization techniques [18–20, 31, 32], we use an improved extreme learning machine [35] to solve SLeNN.

The main contributions of this paper are as below:

- For the first time, a single layer functional link neural network based on shifted Legendre polynomials is established for solving the FDDEs, including constant delay and proportional delay.
- The proposed method is a completely non-iterative neural network method with high efficiency.
- Since the proposed method uses shifted Legendre polynomials in place of the hidden layer of the network, it can perform fractional derivatives of FDDEs analytically without any truncation or numerical discretization.
- Different from the usual neural network methods, the proposed method does not need to initialize the parameters of the neural network randomly, ensuring the stability of the results.

The rest of this paper is arranged as follows. Section 2 briefly introduces the properties and related formulas of fractional calculus and shifted Legendre polynomials. We describe the structure of SLeNN in Section 3. In Section 4, specific methods for solving fractional differential equations with constant and proportional delays are presented, respectively. After that, six numerical examples are given in Section 5 to demonstrate the effectiveness and advantages of our method. Section 6 summarizes the main work of this paper.

2. Mathematical preliminaries

This section first gives some basic definitions and related properties of fractional calculus; see [36] for more details.

Definition 2.1. Let $x(t) \in L^1([t_0, T]; \mathbb{R})$. The Riemann-Liouville fractional integral of order $\alpha > 0$ is defined as

$$ {}^{RL}_{t_0}I^{\alpha}_{t} x(t) = \frac{1}{\Gamma(\alpha)} \int_{t_0}^{t} (t-u)^{\alpha-1} x(u) du, \quad t \in [t_0, T], $$

where $\Gamma(\cdot)$ denotes the Gamma function.

Definition 2.2. Let $x(t) \in AC([t_0, T]; \mathbb{R})$. The Caputo fractional derivative of order $\alpha > 0$ for a function $x(t)$ is defined as

$$ {}^{C}_{t_0}D^{\alpha}_{t} x(t) = \begin{cases} {}^{RL}_{t_0}I^{n-\alpha}_{t} \left(\frac{d}{dx}\right)^n x(t), & n-1 < \alpha < n, \\ \left(\frac{d}{dx}\right)^n x(t), & \alpha = n. \end{cases} $$

Since the derivative of a constant is 0, the following property holds in the Caputo sense:

$$ {}^{C}_{0}D^{\alpha}_{t}(t^k) = \begin{cases} 0, & k \in Z^+, \ k < \lceil \alpha \rceil, \\ \frac{\Gamma(k+1)}{\Gamma(k+1-\alpha)} t^{k-\alpha}, & k \in Z^+, \ k \geq \lceil \alpha \rceil, \ t>0. \end{cases} $$

Next, several basic formulas about shifted Legendre polynomials are presented [37].

Definition 2.3. Shifted Legendre polynomials $Leg_{T,n}(t)$ of degree n have the following explicit analytical form:

$$ Leg_{T,n}(t) = \sum_{s=0}^{n} (-1)^s \frac{(n+s)!(T-t)^s}{T^s (n-s)!(s!)^2} $$

or

$$ Leg_{T,n}(t) = \sum_{s=0}^{n} (-1)^{n+s} \frac{(n+s)! t^s}{T^s (n-s)!(s!)^2}. $$

Property 2.1. The derivative recurrence relation of shifted Legendre polynomials $Leg_{T,n}(t)$ is given by

$$ (2n+1) Leg_{T,n}(t) = \frac{T}{2} \frac{d\left(Leg_{T,n+1}(t) - Leg_{T,n-1}(t)\right)}{dt}, $$

where $n \geq 1$.

3. Shifted Legendre neural network (SLeNN) Model

In this section, we will consider a single layer shifted Legendre neural network model for the present problem. The structure of SLeNN consists of an input neuron, a functional expansion block based on shifted Legendre polynomials and an output neuron. We use shifted Legendre polynomials to expand each input data into several terms. Thus, the shifted Legendre polynomial can be regarded as a new input vector. Let us consider the input neuron t with m training data, which can be denoted as $t = (t_1, t_2, \ldots, t_m)^T$. The basic formulas of shifted Legendre polynomials are given by Definition 2.3 and Property 2.1. It is worth mentioning that the shifted Legendre polynomials satisfy orthogonality on the interval $[0, T]$.

In this work, we augment the input mode by shifted Legendre polynomials. Here, the input vector t is expanded an n-dimensional augmented shifted Legendre polynomial. Then the vector of enhanced mode is obtained by using shifted Legendre polynomials as

$$ [Leg_{T,0}(t_1), \ldots, Leg_{T,n-1}(t_1), Leg_{T,0}(t_2), \ldots, Leg_{T,n-1}(t_2), \ldots, Leg_{T,n-1}(t_m)]. $$

The architecture of the SLeNN with the first five shifted Legendre polynomials, input layer (with single node) and output layer (with single node) is shown in Figure 1. We will describe how to solve the FDDE by SLeNN-ELM in the next section.

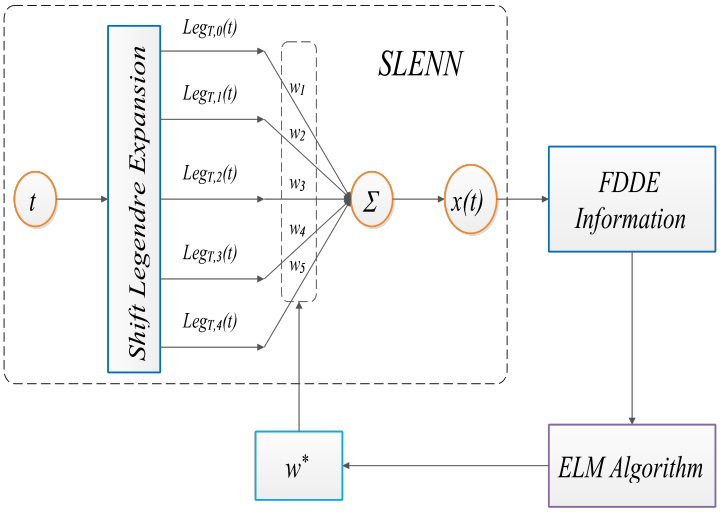

Figure 1. Structure of shift Legendre neural network with $n = 5$ for FDDEs.

4. SLeNN-ELM for solving fractional delay differential equations

In this section, we introduce how to use SLeNN and an improved extreme learning machine to solve fractional differential equations with constant and proportional delays.

At the beginning of this section, for ease of expression, we write the output of SLeNN as the following mathematical expression:

$$\tilde{x}(t) = \sum_{i=1}^{n} w_i * Leg_{T,i-1}(t), \tag{4.1}$$

where w_i represents the connection weight between the shifted Legendre expansion block and the output layer, $\tilde{x}(t)$ is the output of SLeNN.

According to Eq (2.2), we can analytically derive the α-derivative of $\tilde{x}(t)$:

$${}_0^C D_t^\alpha \tilde{x}(t) = \sum_{i=1}^{n} w_i * \frac{\Gamma(i)}{\Gamma(i-\alpha)} Leg_{T,i-1}(t^\alpha). \tag{4.2}$$

4.1. Methodology for solving fractional differential equations with constant delay

We consider the generalized form of fractional differential equations with constant delay, i.e., Eq (1.1). For simplicity, we take $x(t) = \phi(t)$, $t \le 0$ and $\lambda_0 = \phi(0)$ as an example. In the proposed method, we do not need to construct the trial solution, but rather separate the adjustable parameters w_i in SLeNN. Substituting Eqs (4.1) and (4.2) into Eq (1.1) and using simple mathematical

transformations, we can obtain

$$w_i * \left\{ \sum_{i=1}^{n} \frac{\Gamma(i)}{\Gamma(i-\alpha)} Leg_{T,i-1}(t^\alpha) - h_2(t) \sum_{i=1}^{n} Leg_{T,i-1}(t) - h_3(t) \sum_{i=1}^{n} Leg_{T,i-1}(t-\tau) \right\} = h_1(t). \tag{4.3}$$

Now, for different training data of t from the definition domain, we can rewrite the above formula as matrix form $H * w = \beta$ with

$$H = \begin{bmatrix} H_1(t_1) & H_2(t_1) & \dots & H_n(t_1) \\ H_1(t_2) & H_2(t_2) & \dots & H_n(t_2) \\ \vdots & \vdots & & \vdots \\ H_1(t_m) & H_2(t_m) & \dots & H_n(t_m) \end{bmatrix},$$

$$w = (w_1, w_2, \dots, w_n)^T,$$

$$\beta = (h_1(t_1), h_1(t_2), \dots, h_1(t_m))^T,$$

where $H_i(t) = \frac{\Gamma(i)}{\Gamma(i-\alpha)} Leg_{T,i-1}(t^\alpha) - h_2(t) Leg_{T,i-1}(t) - h_3(t) Leg_{T,i-1}(t-\tau)$ and the information about the vector β is already available.

Next, for the initial condition $x(t) = \phi(t), t \le 0$, we take m' training data in $t \le 0$. Likewise, we can construct the matrix form $H_{IC} * w = \beta_{IC}$, where

$$H_{IC} = \begin{bmatrix} Leg_{T,0}(t'_1) & Leg_{T,1}(t'_1) & \dots & Leg_{T,n-1}(t'_1) \\ Leg_{T,0}(t'_2) & Leg_{T,1}(t'_2) & \dots & Leg_{T,n-1}(t'_2) \\ \vdots & \vdots & & \vdots \\ Leg_{T,0}(t'_m) & Leg_{T,1}(t'_m) & \dots & Leg_{T,n-1}(t'_m) \end{bmatrix},$$

$$\beta_{IC} = (\phi(t'_1), \phi(t'_2), \dots, \phi(t'_m))^T.$$

When the solution of SLeNN satisfies both Eq (1.1) and initial conditions, the following linear systems can be obtained:

$$\begin{pmatrix} H \\ H_{IC} \end{pmatrix} w = \begin{pmatrix} \beta \\ \beta_{IC} \end{pmatrix}.$$

Since we cannot guarantee that $\begin{pmatrix} H \\ H_{IC} \end{pmatrix}$ is a non-degenerate or square matrix, we choose to find the minimum norm least square solution of the system, namely:

$$w^* = \begin{pmatrix} H \\ H_{IC} \end{pmatrix}^\dagger \begin{pmatrix} \beta \\ \beta_{IC} \end{pmatrix}, \tag{4.4}$$

where $A^\dagger = \left(A^T A\right)^{-1} A^T$ is the Moore-Penrose generalized inverse and w^* exists and is unique. Finally, we can gain the SLeNN solution by substituting the parameters w_i^* into Eq (4.1) and regard it as the solution of FDDEs.

4.2. Methodology for solving fractional differential equations with proportional delay

Here, we consider the fractional differential equations with proportional delay, i.e., Eq (1.2). To make it easy, let's take $x(0) = \lambda_0$ and $x'(0) = \lambda_1$ as an example. Similar to Section 4.1, we substitute Eqs (4.1) and (4.2) into Eq (1.2) and move all terms related to x to the left side of the formula,

$$w_i * \left\{ \sum_{i=1}^{n} \frac{\Gamma(i)}{\Gamma(i-\alpha)} Leg_{T,i-1}(t^\alpha) - g_2(t) \sum_{i=1}^{n} Leg_{T,i-1}(t) - g_3(t) \sum_{i=1}^{n} Leg_{T,i-1}(\tau t) \right\} = g_1(t). \tag{4.5}$$

Now, in order to get the matrix form $G * w = \beta$, we take m number of training points from the definition domain. By exploiting these training points, we have

$$G = \begin{bmatrix} G_1(t_1) & G_2(t_1) & \dots & G_n(t_1) \\ G_1(t_2) & G_2(t_2) & \dots & G_n(t_2) \\ \vdots & \vdots & & \vdots \\ G_1(t_m) & G_2(t_m) & \dots & G_n(t_m) \end{bmatrix},$$

$$w = (w_1, w_2, \dots, w_n)^T,$$

$$\beta = (g_1(t_1), g_1(t_2), \dots, g_1(t_m))^T,$$

with $G_i(t) = \frac{\Gamma(i)}{\Gamma(i-\alpha)} Leg_{T,i-1}(t^\alpha) - g_2(t) Leg_{T,i-1}(t) - g_3(t) Leg_{T,i-1}(t-\tau)$.

Then we consider two initial conditions $x(0) = \lambda_0$ and $x'(0) = \lambda_1$. In order to make the obtained SLeNN-ELM solution satisfy the initial conditions, we also need to construct the matrix form of $G_{IC} * w = \beta_{IC}$ as follows:

$$\begin{bmatrix} Leg_{T,0}(0) & Leg_{T,1}(0) & \dots & Leg_{T,n-1}(0) \\ 0 & Leg_{T,1}(0) & \dots & (n-1)Leg_{T,n-1}(0) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} \lambda_0 \\ \lambda_1 \end{bmatrix}.$$

After that, the above problem is solved using the Moore-Penrose generalized inverse theory. Similar to Eq (4.4), we gain the parameters of SLeNN by the following formula

$$w^* = \begin{pmatrix} G \\ G_{IC} \end{pmatrix}^\dagger \begin{pmatrix} \beta \\ \beta_{IC} \end{pmatrix}. \tag{4.6}$$

Since the target matrix is not square in most cases (depending on the number of training points and the order of the SLeNN), we have to use a pseudo-inverse. The obtained optimal parameters can be used to represent the established SLeNN. Notably, unlike traditional gradient-based neural network training methods, this method does not require any optimization process.

5. Numerical illustrations

In this section, we use six numerical experiments to verify the accuracy and superiority of the proposed method. All experiments are performed on a computer, which is configured as follows: Intel(R) Core(TM) i5-8265U CPU @ 1.60 GHz 1.80 GHz.

5.1. Fractional differential equations with constant delay

Example 5.1. Consider the following fractional differential equations with constant delay [27, 38]:

$$
\begin{aligned}
&{}_0^C D_t^{0.2} x(t) = -x(t-1) + \frac{\Gamma(3)}{\Gamma(2.8)} t^{1.8} - \frac{\Gamma(2)}{\Gamma(1.8)} t^{0.8} + t^2 - 3t + 1, \quad t > 0, \\
&x(t) = t^2 - t - 1, \quad -1 \le t \le 0.
\end{aligned}
$$

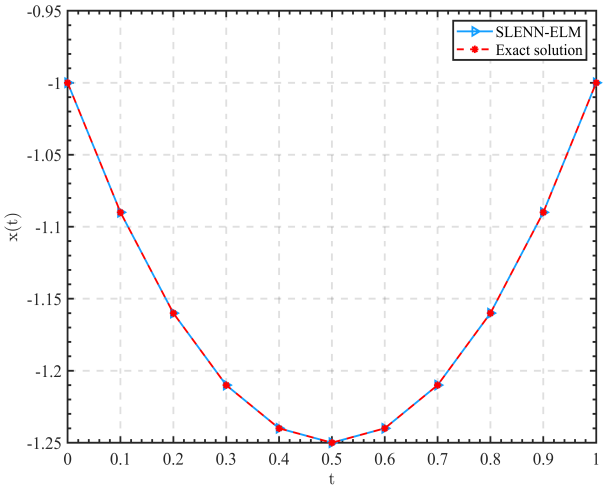

Figure 2. The SLeNN-ELM solution and the exact solution for Example 5.1.

The exact solution is t^2–t–1. For a more visual comparison, we show the curves of the SLeNN-ELM solution and the exact solution in Figure 2, with the SLeNN-ELM solution in blue and the exact solution in red. Table 1 shows the comparison of the absolute errors of the proposed method and Legendre wavelet method [27] at different model parameters. The results show that our method outperforms the Legendre wavelet method. It can be seen that our method is very effective, with errors as low as $O(10^{-16})$.

Table 1. The absolute errors for Example 5.1 by using Legendre wavelet method [27] with $M = 2$, $k = 7$, Legendre wavelet method with $M = 3$, $k = 7$, our method with $m = 3$, $n = 10$ and our method with $m = 6$, $n = 10$.

t	Legendre wavelet $M = 2, k = 7$	Legendre wavelet $M = 3, k = 7$	our method $m = 3, n = 10$	our method $m = 6, n = 10$
0.2	9.28×10^{-6}	7.66×10^{-7}	4.44×10^{-16}	2.22×10^{-16}
0.4	1.11×10^{-6}	4.37×10^{-7}	6.66×10^{-16}	4.44×10^{-16}
0.6	3.38×10^{-6}	3.12×10^{-7}	8.88×10^{-16}	4.44×10^{-16}
0.8	1.04×10^{-6}	2.50×10^{-7}	8.88×10^{-16}	2.22×10^{-16}
1	1.70×10^{-6}	2.03×10^{-7}	1.11×10^{-15}	2.22×10^{-16}

Example 5.2. We consider the following FDDE [38]:

$$
\begin{aligned}
&{}_0^C D_t^{0.4} x(t) + {}_0^C D_t^{0.3} x(t) + x(t-1) = \frac{\Gamma(3)t^{1.6}}{\Gamma(2.6)} - \frac{\Gamma(2)t^{0.6}}{\Gamma(1.6)} + \frac{\Gamma(3)t^{1.7}}{\Gamma(2.7)} - \frac{\Gamma(2)t^{0.7}}{\Gamma(1.7)} \\
&\qquad\qquad + t^2 - 3t + 2, \quad t \in [0,1], \\
&x(t) = t^2 - t, \quad t \in [-1,0], \\
&x(0) = x(1).
\end{aligned}
$$

Here, we have $x(t) = t^2 - t$ as the exact solution. We plot the predicted solution obtained by SLeNN-ELM and compare it with the exact solution in Figure 3. What can be observed is that the SLeNN-ELM method exhibits good accuracy. Table 2 compares the absolute errors of the exact solutions of our method and the method in [38]. The root mean squared error and the max absolute error of our method are 1.04×10^{-15} and 1.33×10^{-15}, respectively.

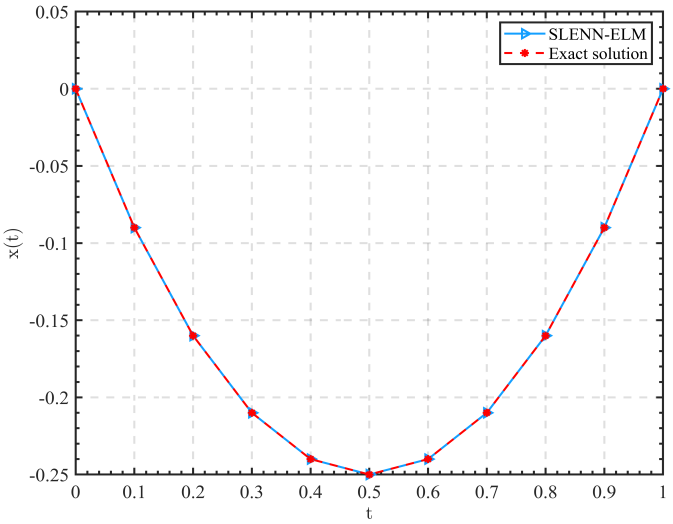

Figure 3. The SLeNN-ELM solution and the exact solution for Example 5.2.

Table 2. The absolute errors for Example 5.2 by using method in [38] with $h = 0.0005$ and our method with $m = 6$, $n = 10$.

t	method in [38] $h = 0.0005$	our method $m = 6, n = 10$
0.2	10^{-7}	9.44×10^{-16}
0.4	10^{-7}	9.44×10^{-16}
0.6	10^{-7}	1.11×10^{-15}
0.8	10^{-7}	1.03×10^{-15}
1	10^{-7}	1.33×10^{-15}

Example 5.3. In this example, we consider the following equation with constant delay [6, 41, 42]:

$$
\begin{aligned}
&{}_0^C D_t^{\alpha} x(t) + x(t) + x(t-0.3) = e^{-t+0.3}, \quad t \geq 0, \quad 2 < \alpha \leq 3,\\
&x(t) = e^{-t}, \quad -0.3 \leq t \leq 0,\\
&x(0) = -x'(0) = x''(0) = 1.
\end{aligned}
$$

Table 3. The absolute errors for Example 5.3 by using Chebyshev wavelets method [41], method in [6] and our method with $m = 10$, $n = 10$.

t	Chebyshev wavelets method	method in [6]	our method
0.01	8.20×10^{-9}	4.53×10^{-9}	2.61×10^{-13}
0.02	6.68×10^{-8}	1.84×10^{-9}	1.09×10^{-13}
0.03	2.29×10^{-7}	2.81×10^{-9}	1.04×10^{-12}
0.04	5.51×10^{-7}	4.46×10^{-10}	2.74×10^{-12}
0.05	1.09×10^{-6}	5.79×10^{-10}	5.34×10^{-12}
0.06	1.91×10^{-6}	3.10×10^{-10}	8.95×10^{-12}
0.07	3.09×10^{-6}	2.14×10^{-10}	1.37×10^{-11}
0.08	4.67×10^{-6}	8.79×10^{-10}	1.95×10^{-11}
0.09	6.75×10^{-6}	1.60×10^{-9}	2.66×10^{-11}
0.10	9.40×10^{-6}	2.32×10^{-9}	3.48×10^{-11}

When $\alpha = 3$, the exact solution is e^{-t}. To see the characteristic of the solution more intuitively, we show in Figure 4 the resulting SLeNN-ELM solution and the exact solution, which are two curves that overlap almost precisely. In Table 3, we list the results of Chebyshev wavelets method [41], method in [6] and our method with $m = 10$, $n = 10$. Here, in order to match and compare the results reported in the literature above, we only present the absolute error value at $t \in [0, 0.1]$. We observe that the proposed method is more accurate than other methods.

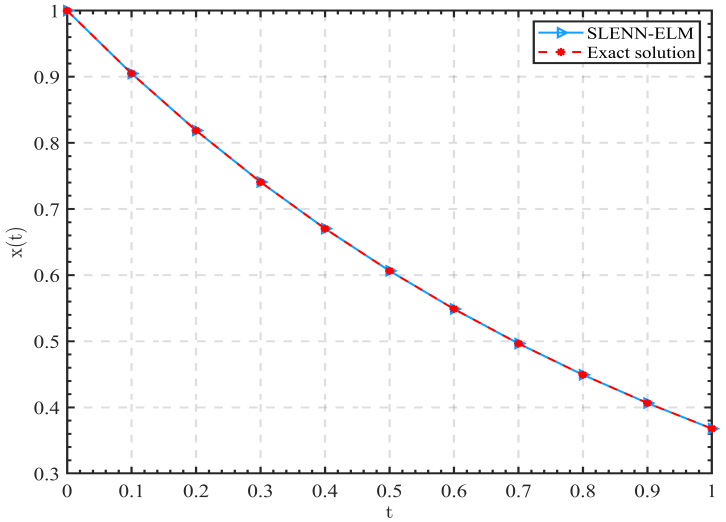

Figure 4. The SLeNN-ELM solution and the exact solution for Example 5.3.

Moreover, to demonstrate the effectiveness of our method more visually, we depict the absolute error by our method for Example 5.3 in Figure 5. From Figure 6, we can find that the solutions obtained by our method vary consistently from fractional to integer order, indicating that the fractional order solutions are accurate.

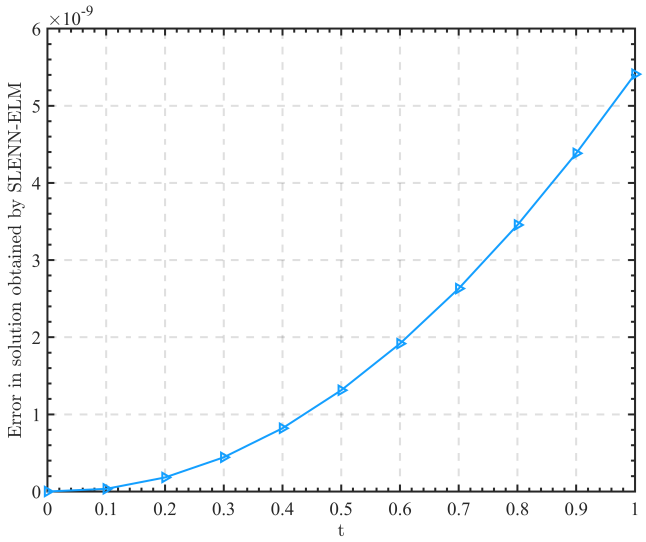

Figure 5. Error in obtained SLeNN-ELM solutions for Example 5.3 by using $m = 10$, $n = 10$.

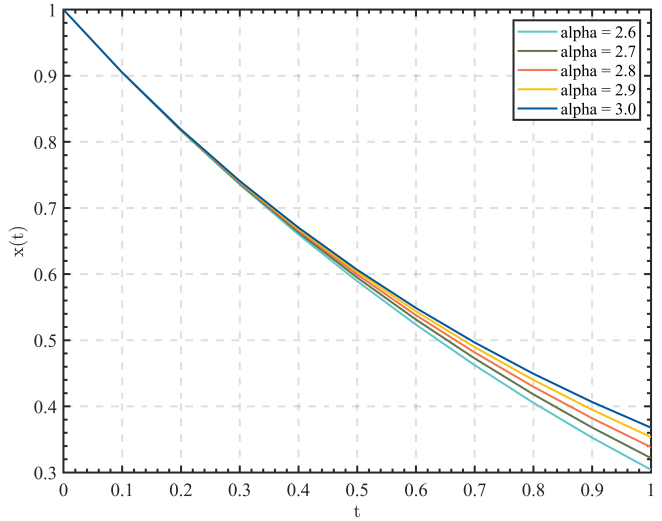

Figure 6. The SLeNN-ELM solution for Example 5.3 at various values of α.

5.2. Fractional differential equations with proportional delay

Example 5.4. Consider the following example of fractional differential equations with proportional delay [14, 27]:

$$ {}_0^C D_t^\alpha x(t) = -x(t) + \frac{1}{10} x\left(\frac{4}{5}t\right) + \frac{1}{2} {}_0^C D_t^\alpha x\left(\frac{4}{5}t\right) + \left(\frac{8}{25}t - \frac{1}{2}\right) e^{-\frac{4}{5}t} + e^{-t}, $$

where $t \in [0, 1]$ and $\alpha \in (0, 1]$ with the initial condition $x(0) = 0$.

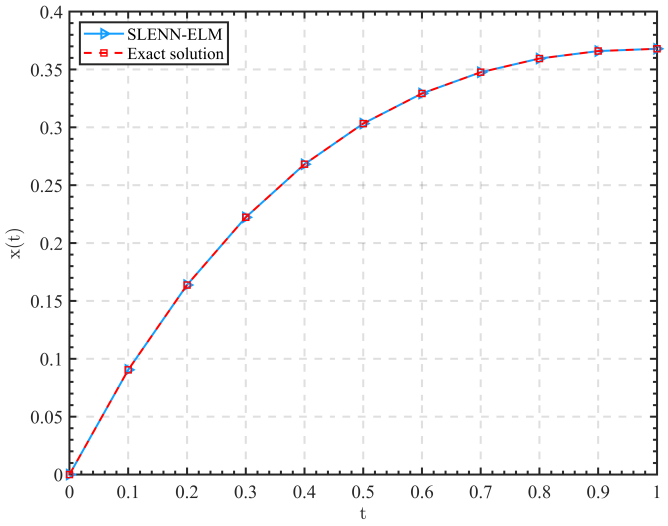

Figure 7. The SLeNN-ELM solution and the exact solution for Example 5.4.

When $\alpha = 1$, we get $x(t) = te^{-t}$ as the exact solution. We plot the solution obtained by SLeNN-ELM and the exact solution in Figure 7 to facilitate our observation of the characteristics of the solution. To highlight the advantages of the proposed method, we have also compared it with other numerical methods. For $\alpha = 1$, Table 4 shows the absolute errors of our method with $m = 10$, $n = 10$ and other methods: variational iteration method [39] with $\hat{m} = 6$, two-stage order-one Runge-Kutta method [40], generalized fractional-order Bernoulli wavelet [14] with $k = 2$, $M1 = 6$, $\alpha = 1$ and Legendre wavelet method [27] with $k = 2$, $M = 6$.

Table 4. The absolute errors for Example 5.4 by using variational iteration method [39] with $\hat{m} = 6$, two-stage order-one Runge-Kutta method [12], generalized fractional-order Bernoulli wavelet [14] with $k = 2$, $M1 = 6$, $\alpha = 1$, Legendre wavelet method [27] with $k = 2$, $M = 6$ and our method with $m = 10$, $n = 10$.

t	variational iteration	Runge-Kutta	Bernoulli wavelet	Legendre wavelet	our method
0.1	1.30×10^{-3}	8.68×10^{-4}	1.98×10^{-8}	9.76×10^{-9}	5.44×10^{-11}
0.3	2.63×10^{-3}	1.90×10^{-3}	7.78×10^{-9}	5.67×10^{-9}	3.89×10^{-11}
0.5	2.83×10^{-3}	2.28×10^{-3}	6.34×10^{-5}	7.75×10^{-9}	2.53×10^{-11}
0.7	2.39×10^{-3}	2.27×10^{-3}	4.36×10^{-5}	6.91×10^{-9}	1.67×10^{-11}
0.9	1.64×10^{-3}	2.03×10^{-3}	2.80×10^{-5}	5.57×10^{-9}	1.37×10^{-11}

When $\alpha \in (0, 1)$, the exact solution is unknown. To prove the accuracy of the proposed method, we extend the domain from $t \in [0, 1]$ to $t \in [0, 8]$ and plot the SLeNN-ELM solution at different values of α in Figure 8. The obtained results are in good agreement with those in [27].

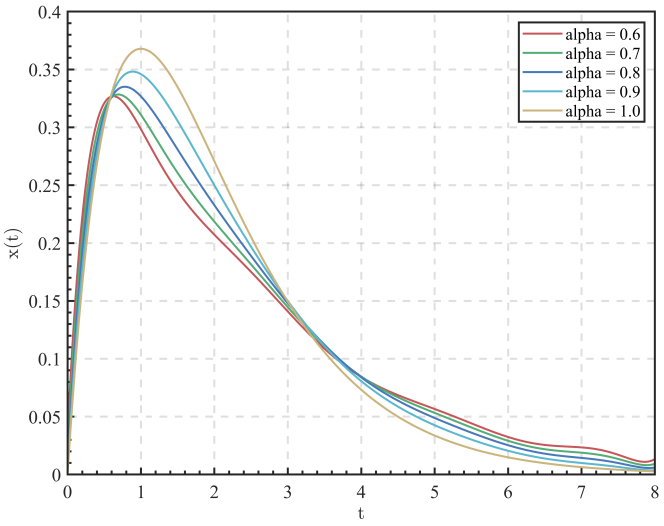

Figure 8. The SLeNN-ELM solution for Example 5.4 at various values of α with $m = 10$, $n = 10$.

Example 5.5. As the second example, consider [43, 44]:

$$
\begin{aligned}
&{}_0^C D_t^{0.5} x(t) = -x(t) + x\left(\frac{t}{2}\right) + \frac{7}{8}t^3 + \frac{16}{5\Gamma(0.5)}t^{2.5}, \\
&x(0) = 0,
\end{aligned}
$$

and the exact solution for this example is t^3.

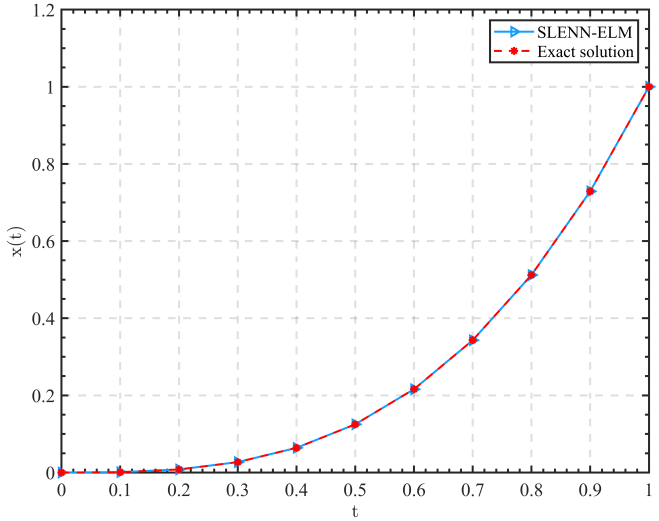

Figure 9. The SLeNN-ELM solution and the exact solution for Example 5.5.

In Figure 9, we can visualize the accuracy of the SLeNN-ELM solution by plotting the exact solution and the neural network solution. The comparison of absolute values of errors at grid points selected by our method and method in [44] is shown in Table 5. Here, the parameters $m = 8$, $n = 10$ are chose. The error obtained by our method is at least $O(10^{-16})$. Obviously, our method performs better.

Table 5. The absolute errors for Example 5.5 by using method in [44] with $N = 12$ and our method with $m = 8$, $n = 10$.

t	method in [44], $N = 12$	our method, $m = 8$, $n = 10$
0.1	2.21×10^{-7}	5.62×10^{-17}
0.2	9.89×10^{-7}	5.38×10^{-17}
0.3	3.41×10^{-6}	1.11×10^{-16}
0.4	1.95×10^{-5}	6.94×10^{-16}
0.5	5.62×10^{-5}	1.94×10^{-16}
0.6	1.20×10^{-4}	5.55×10^{-17}
0.7	2.25×10^{-4}	0
0.8	3.75×10^{-4}	0
0.9	5.80×10^{-4}	2.22×10^{-16}
1.0	8.48×10^{-4}	2.22×10^{-16}

Example 5.6. We give a higher-order example of proportional delay. Consider the equation as follows:

$$ {}_0^C D_t^{1.7} x(t) = x(t) - 3x(\frac{t}{2}) + \frac{\Gamma(3)}{\Gamma(1.3)} t^{0.3} - \frac{1}{4} t^2, $$

with the initial condition $x(0) = x'(0) = 0$ and we obtain $x(t) = t^2$ as the exact solution.

Table 6. The absolute errors for Example 5.6 by using our method with $m = 5$, $n = 10$ and $m = 7$, $n = 10$.

t	our method $m = 5$, $n = 10$	our method $m = 7$, $n = 10$
0.1	2.14×10^{-15}	1.99×10^{-15}
0.2	2.26×10^{-15}	1.89×10^{-15}
0.3	2.36×10^{-15}	1.55×10^{-15}
0.4	2.35×10^{-15}	1.36×10^{-15}
0.5	2.44×10^{-15}	9.99×10^{-16}
0.6	2.55×10^{-15}	7.21×10^{-16}
0.7	2.50×10^{-15}	5.00×10^{-16}
0.8	2.33×10^{-15}	5.55×10^{-16}
0.9	2.22×10^{-15}	2.22×10^{-16}
1.0	2.22×10^{-15}	0

The exact solution and the SLeNN-ELM solution are graphed in Figure 10. It can be observed that the two curves are almost perfectly fitted. In Table 6, we show the absolute values of the error using

our method with $m = 5$, $n = 10$ and $m = 7$, $n = 10$, respectively. We note that different parameter values both provide a fairly high accuracy for our SLeNN-ELM solutions.

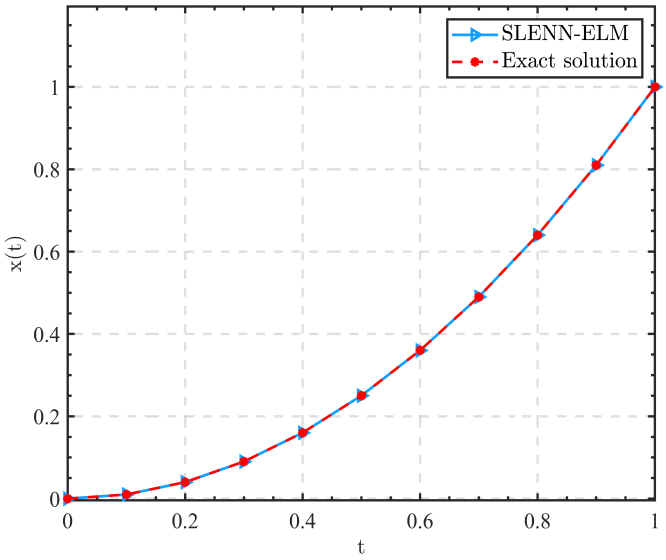

Figure 10. The SLeNN-ELM solution and the exact solution for Example 5.6.

Finally, we list the CPU time required by the proposed and other methods in Table 7. We observe that the proposed method can achieve good accuracy with very high efficiency.

Table 7. CPU time required for the proposed and other methods in all examples.

Example	Method	Selected parameters	CPU time (s)
5.1	Legendre wavelet method [27]	M = 2, k = 7	41.5627
	our method	m = 6, n = 10	1.0754
5.2	method in [38]	h = 0.0005	69.3673
	our method	m = 6, n = 10	1.0216
5.3	method in [6]	n = 9	45.1560
	our method	m = 10, n = 10	0.7336
5.4	Legendre wavelet method [27]	M = 6, k = 2	6.5233
	our method	m = 10, n = 10	0.5683
5.5	method in [44]	N = 12	38.7851
	our method	m = 8, n = 10	0.4590
5.6	our method	m = 7, n = 10	0.4053

6. Conclusion

In the present work, we propose a shifted Legendre neural network based on an extreme learning machine algorithm (SLeNN-ELM) to solve fractional differential equations with constant and proportional delays. Neural network methods have many advantages over classical numerical

methods in solving differential equations. For example, the solution obtained is analytical in essence. This is helpful for us to better study the properties of the solution of the equation. Different from the traditional neural network method, we use the extreme learning machine algorithm instead of the gradient-based training process, which significantly improves the computational efficiency under the premise of ensuring the accuracy of the solution. In addition, SLeNN is a FLNN, which uses function expansion blocks instead of hidden layers to improve the computational efficiency of the network by reducing the number of parameters. The required calculation time for our method is listed in Table 7. On the other hand, the proposed method can perform fractional derivatives analytically without any truncation or numerical discretization for the fractional problem. More importantly, the proposed method does not need to initialize parameters randomly like other neural network methods, which enables us to obtain a stable solution to any problem. Finally, six examples are given, and the proposed method is compared with other numerical methods. The numerical results show the effectiveness and superiority of our method. In future work, we will consider combining multi-hidden layer neural networks [45] and extreme learning machine algorithms to better solve fractional differential equations with different time delays.

Acknowledgments

The authors express their sincere thanks to the anonymous reviewer for his/her careful reading of the paper, giving valuable comments and suggestions. It is their contributions that greatly improve the paper. The authors also thank the editors for their kind help.

This work was supported by the National Natural Science Foundation of China (Nos. 11801452, 11701456), Fundamental Research Project of Natural Science in Shaanxi Province General Project (Youth) (Nos.2019JQ-415, 2019JQ-196), Chinese Universities Scientific Funds(Nos. 2452022104, 2452022105, 2452022372), and Innovation and Entrepreneurship Training Program for College Students of Northwest A&F University (202110712186, 202210712048, S202210712377).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. C. T. Baker, C. A. Paul, D. R. Willé, Issues in the numerical solution of evolutionary delay differential equations, *Adv. Comput. Math*, **3** (1995), 171–196. https://doi.org/10.1007/BF02988625
2. R. D. Driver, *Ordinary and delay differential equations*, New York: Springer, 2012.
3. Y. Kuang, *Delay Differential Equations with Applications in Population Dynamics*, New York: Academic Press, 1993.
4. J. N. Luo, W. H. Tian, S. M. Zhong, K. B. Shi, X. M. Gu, W. Q. Wang, Improved delay-probability-dependent results for stochastic neural networks with randomly occurring uncertainties and multiple delays, *Int. J. Syst. Sci.*, **49** (2018), 2039–2059. https://doi.org/10.1080/00207721.2018.1483044

5. H. Singh, A new numerical algorithm for fractional model of Bloch equation in nuclear magnetic resonance, *Alex. Eng. J.*, **55** (2016), 2863–2869. https://doi.org/10.1016/j.aej.2016.06.032
6. H. Singh, Numerical simulation for fractional delay differential equations, *Int. J. Dyn. Control*, **9** (2021), 463–474. https://doi.org/10.1007/s40435-020-00671-6
7. J. R. Ockendon, A. B. Tayler, The dynamics of a current collection system for an electric locomotive, *Proc. R. Soc. Lond. Ser. A*, **322** (1971), 447–468. https://doi.org/10.1098/rspa.1971.0078
8. V. Kolmanovskii, A. Myshkis, *Introduction to the Theory and Applications of Functional Differential Equations*, Berlin: Springer Science & Business Media, 1999.
9. U. Farooq, H. Khan, D. Baleanu, M. Arif, Numerical solutions of fractional delay differential equations using Chebyshev wavelet method, *J. Comput. Appl. Math.*, **38** (2019), 1–13. https://doi.org/10.1007/s40314-019-0953-y
10. M. L. Morgado, N. J. Ford, P. M. Lima, Analysis and numerical methods for fractional differential equations with delay, *J. Comput. Appl. Math.*, **252** (2013), 159–168. https://doi.org/10.1016/j.cam.2012.06.034
11. S. Bhalekar, S. Daftardar-Gejji, A predictor-corrector scheme for solving nonlinear delay differential equations of fractional order, *J. Fract. Calc. Appl.*, **1** (2011), 1–9.
12. W. Wang, Y. Zhang, S. Li, Stability of continuous Runge-Kutta-type methods for nonlinear neutral delay-differential equations, *Appl. Math. Model.*, **33** (2009), 3319–3329. https://doi.org/10.1016/j.apm.2008.10.038
13. B. P. Moghaddam, Z. S. Mostaghim, A numerical method based on finite difference for solving fractional delay differential equations, *J. Taibah Univ. Sci.*, **7** (2013), 120–127. https://doi.org/10.1016/j.jtusci.2013.07.002
14. P. Rahimkhani, Y. Ordokhani, E. Babolian, A new operational matrix based on Bernoulli wavelets for solving fractional delay differential equations, *Numer. Algorithms*, **74** (2017), 223–245. https://doi.org/10.1007/s11075-016-0146-3
15. R. K. Pandey, N. Kumar, R. Mohaptra, An approximate method for solving fractional delay differential equations, *Int. J. Appl. Comput. Math.*, **3** (2017), 1395–1405. https://doi.org/10.1007/s40819-016-0186-3
16. U. Saeed, M. Rehman, M. A. Iqbal, Modified Chebyshev wavelet methods for fractional delay-type equations, *Appl. Math. Comput.*, **264** (2015), 431–442. https://doi.org/10.1016/j.amc.2015.04.113
17. S. Panghal, M. Kumar, Optimization free neural network approach for solving ordinary and partial differential equations, *Eng. Comput.*, **37** (2021), 2989–3002. https://doi.org/10.1007/s00366-020-00985-1
18. A. Jafarian, M. Mokhtarpour, D. Baleanu, Artificial neural network approach for a class of fractional ordinary differential equation, *Neural Comput. Appl.*, **28** (2017), 765–773. https://doi.org/10.1007/s00521-015-2104-8
19. I. E. Lagaris, A. Likas, D. I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, *IEEE Trans. Neural Netw.*, **9** (1998), 987–1000. https://doi.org/10.1109/72.712178

20. A. Jafarian, S. M. Nia, A. K. Golmankhaneh, D. Baleanu, On artificial neural networks approach with new cost functions, *Appl. Math. Comput.*, **339** (2018), 546–555. https://doi.org/10.1016/j.amc.2018.07.053

21. C. C. Hou, T. E. Simos, I. T. Famelis, Neural network solution of pantograph type differential equations, *Math. Method. Appl. Sci.*, **43** (2020), 3369–3374. https://doi.org/10.1002/mma.6126

22. B. Shiri, H. Kong, G. C. Wu, C. Luo, Adaptive learning neural network method for solving time-fractional diffusion equations, *Neural Comput.*, **34** (2022), 971–990. https://doi.org/10.1162/neco_a_01482

23. Y. Ye, H. Fan, Y. Li, X. Liu, H. Zhang, Deep neural network methods for solving forward and inverse problems of time fractional diffusion equations with conformable derivative, *Neurocomputing*, **509** (2022), 177–192. https://doi.org/10.1016/j.neucom.2022.08.030

24. C. D. Huang, H. Liu, X. Y. Shi, X. P. Chen, M. Xiao, Z. X. Wang, et al., Bifurcations in a fractional-order neural network with multiple leakage delays, *Neural Netw.*, **131** (2020), 115–126. https://doi.org/10.1016/j.neunet.2020.07.015

25. C. J. Xu, D. Mu, Z. X. Liu, Y. C. Pang, M. X. Liao, P. L. Li, et al., Comparative exploration on bifurcation behavior for integer-order and fractional-order delayed BAM neural networks, *NONLINEAR ANAL-MODEL*, **27** (2022), 1030–1053. https://doi.org/10.15388/namc.2022.27.28491

26. C. J. Xu, Z. X. Liu, C. Aouiti, P. L. Li, L. Y. Yao, J. L. Yan, New exploration on bifurcation for fractional-order quaternionvalued neural networks involving leakage delays, *Cogn. Neurodyn.*, **16** (2020), 1233–1248. https://doi.org/10.1007/s11571-021-09763-1

27. B. Yuttanan, M. Razzaghi, T. N. Vo, Legendre wavelet method for fractional delay differential equations, *Appl. Numer. Math.*, **168** (2021), 127–142. https://doi.org/10.1016/j.apnum.2021.05.024

28. H. Marzban, M. Razzaghi, Hybrid functions approach for linearly constrained quadratic optimal control problems, *Appl. Math. Model.*, **27** (2003), 471–485. https://doi.org/10.1016/S0307-904X(03)00050-7

29. S. S. Yang, C. S. Tseng, An orthogonal neural network for function approximation, *IEEE Trans. Syst. Man Cybern.*, **26** (1996), 779–785. https://doi.org/10.1109/3477.537319

30. A. Saadatmandi, M. Dehghan, A new operational matrix for solving fractional-order differential equations, *Comput. Math. Appl.*, **59** (2010), 1326–1336. https://doi.org/10.1016/j.camwa.2009.07.006

31. H. D. Qu, Z. H. She, X. Liu, Neural network method for solving fractional diffusion equations, *Appl. Math. Comput.*, **391** (2021), 125635. https://doi.org/10.1016/j.amc.2020.125635

32. H. D. Qu, X. Liu, X. Lu, M. ur Rahman, Z. H. She, Neural network method for solving nonlinear fractional advection-diffusion equation with spatiotemporal variable-order, *Chaos Solitons Fractals*, **156** (2022), 111856. https://doi.org/10.1016/j.chaos.2022.111856

33. J. C. Patra, P. K. Meher, G. Chakraborty, Nonlinear channel equalization for wireless communication systems using Legendre neural networks, *Signal Process*, **89** (2009), 2251–2262. https://doi.org/10.1016/j.sigpro.2009.05.004

34. J. C. Patra, C. Bornand, Nonlinear dynamic system identification using Legendre Neural Network, *The 2010 international joint conference on neural networks (IJCNN)*, (2010). https://doi.org/10.1109/IJCNN.2010.5596904

35. G. B. Huang, Q. Zhu, C. K. Siew, Extreme learning machine: theory and applications, *Neurocomputing*, **70** (2006), 489–501. https://doi.org/10.1016/j.neucom.2005.12.126

36. A. A. Kilbsa, H. M. Srivastava, J. J. Trujillo, *Theory and Applications of Fractional Differential Equations*, New York: Elsevier, 2006.

37. J. Shen, T. Tang, L. L. Wang, *Spectral Methods: Algorithms, Analysis and Applications*, Berlin: Springer, 2011.

38. M. S. Heris, M. Javidi, On fractional backward differential formulas for fractional delay differential equations with periodic and anti-periodic conditions, *Appl. Numer. Math.*, **118** (2017), 203–220. https://doi.org/10.1016/j.apnum.2017.03.006

39. X. Chen, L. Wang, The variational iteration method for solving a neutral functional-differential equation with proportional delays, *Comput. Math. Appl.*, **59** (2010), 2696–2702. https://doi.org/10.1016/j.camwa.2010.01.037

40. A. Bellen, M. Zennaro, *Numerical Methods for Delay Differential Equations*, Oxford: Oxford University Press, 2013.

41. M. A. Iqbal, A. Ali, S. T. Mohyud-Din, Chebyshev wavelets method for fractional delay differential equations, *Int. J. Mod. Appl. Phys.*, **4** (2013), 49–61.

42. M. A. Iqbal, U. Saeed, S. T. Mohyud-Din, Modified Laguerre wavelets method for delay differential equation of fractional-order, *Egypt. J. Bas. Appl. Sci.*, **2** (2015), 50–54. https://doi.org/10.1016/j.ejbas.2014.10.004

43. M. Dehghan, S. A. Yousefi, A. Lotfi, The use of He's variational iteration method for solving the telegraph and fractional telegraph equations, *Int. J. Numer. Methods Biomed. Eng.*, **27** (2011), 219–231. https://doi.org/10.1002/cnm.1293

44. M. Ghasemi, M. Fardi, R. K. Ghaziani, Numerical solution of nonlinear delay differential equations of fractional order in reproducing kernel Hilbert space, *Appl. Math. Comput.*, **268** (2015), 815–831. https://doi.org/10.1016/j.amc.2015.06.012

45. J. L. Wei, G. C. Wu, B. Q. Liu, Z. G. Zhao, New semi-analytical solutions of the time-fractional Fokker-Planck equation by the neural network method, *Optik*, **259** (2022), 168896. https://doi.org/10.1016/j.ijleo.2022.168896

AIMS Press