Available online at www.sciencedirect.com

ScienceDirect

Mathematics and Computers in Simulation 208 (2023) 15–27

MATHEMATICS AND COMPUTERS IN SIMULATION

www.elsevier.com/locate/matcom

Original articles

Study of fractional-order reaction-advection-diffusion equation using neural network method

Chetna Biswas[a], Anup Singh[b], Manish Chopra[c], Subir Das[a,*]

[a] *Department of Mathematical Sciences, Indian Institute of Technology (BHU), Varanasi 221005, India*
[b] *Department of Mathematics, Institute of Technology, Nirma University, Ahmedabad 382481, Gujarat, India*
[c] *Radiation Safety Systems Division, Bhabha Atomic Research Centre, Mumbai 400085, India*

Received 20 November 2022; received in revised form 21 December 2022; accepted 30 December 2022
Available online 21 January 2023

Abstract

In the present article the spatio-temporal fractional-order nonlinear reaction-advection-diffusion equation is solved using the neural network method (NNM). Shifted Legendre orthogonal polynomials with variable coefficients are used in the network's construction. The characteristics of a fractional-order derivative are used to determine the loss function of a neural network. The permissible learning rate range is discussed in detail, assuming that the Lipschitz hypothesis is accurate for the nonlinearity in reaction term. We have demonstrated the application of the NNM on two numerical examples by utilizing the neural networks which had been repeatedly trained on the training set. In other words, we have validated the effectiveness of the method for such problems. The effects of reaction term and also the degree of nonlinearity in reaction and advection terms on the solution profile are visualized through graphical presentations for specific test cases.
© 2023 International Association for Mathematics and Computers in Simulation (IMACS). Published by Elsevier B.V. All rights reserved.

Keywords: Fractional-order reaction-advection-diffusion equation; Neural network method; Artificial neural network loss function; Damping

1. Introduction

The fractional order integrals and derivatives were developed practically at the same time when the integer calculus was developed. Fractional calculus has attracted a lot of attention recently due to its wide applications in a variety of fields including financial systems, physical, chemical, geological and biological systems [3,4]. No approach provides an exact solution to the fractional-order differential equation (FDE). However, the nonlinear fractional order partial differential equations (FPDEs) have attracted special attentions of the scientific community [2,5,11,16,19,20,24,25]. The time memory or historical inheritance of a fractional derivative is its most notable characteristic [9], because of which, fractional derivatives have extensive applications in diverse domains [10]. FPDE can be used to represent a variety of natural systems, including the thermal pollution of river systems, atmospheric pollution and groundwater pollution. The equation that describes the flows in porous media is solved to determine the velocities of the transport medium. Although the flow equations are nonlinear, diffusion and advection are the most crucial factors. The advection–diffusion equation describes how a solute is transported when advection

* Corresponding author.
E-mail address: sdas.apm@iitbhu.ac.in (S. Das).

https://doi.org/10.1016/j.matcom.2022.12.032
0378-4754/© 2023 International Association for Mathematics and Computers in Simulation (IMACS). Published by Elsevier B.V. All rights reserved.

and diffusion are acting together. reaction-advection-diffusion equations, another type of chemical equation, are shown to exist when the chemical being carried through soil is reactive. In general, fractional differential models do not have any analytical solution, or if they do, the analytical solutions are challenging to calculate because of the involvement of many complicated functions in such models. Therefore, it is crucial to put more effort in research related to FDEs for their numerical solutions. Nonlinear FDEs are typically difficult to solve precisely, necessitating the use of numerical methods and methods of approximation. Researchers in this field have primarily suggested three numerical procedures to effectively solve the corresponding FDEs: the finite element method (FEM), the spectrum method (SM) and the finite difference method (FDM). In order to solve various types of linear and nonlinear ordinary FDEs, Raja et al. [14] have developed a feed-forward artificial neural network (ANN)-based method. He demonstrated the usefulness of this approach along with the limitation that this approach provides less accuracy while dealing with difficult nonlinear FDEs.

The ANN based approach for numerical solutions to FDEs was examined by [12] to show the effectiveness of these networks. The effectiveness of this technique was demonstrated by comparing with the analytical solution and a number of numerical techniques which are currently in use. Improvement of the performance in thermal and environmental processes using ANN with conformable transfer function is done in [22]. An overview of real-world uses for ANN in fractional calculus can be found in [26]. The trial solutions of models are built from a combination of adjustable and non-adjustable elements. PDEs, linked with ODE systems, and individual ODEs can be solved using the said technique. In [6], the authors thought about using an iterative approach to solve FDEs and used the generalized sigmoid function as the cost function. Wei et al. [27] suggested that the time-fractional Fokker–Planck equation can be solved using a neural network. In [8], a complete description of radial basis function in neural network algorithms to solve various kinds of differential equations has been provided. The sequential quadratic programming (SQP) algorithm is used to effectively update the weights of the network via restricted optimization. In [15], the authors have created a new computing method employing fractional neural networks to solve fractional-order Bagley–Torvik equations with initial conditions. In recent years, it is seen a growth of crucial dynamical problems, dependent on time, space, or both, showing behaviour of the fractional-order. The fractional Adams–Bashforth approach was described in its right form in [1]. Functional link neural network (FLNN), a higher order neural network, was used to represent linear and nonlinear delay fractional optimal control problems (DFOCPs) with mixed control-state constraints in [7]. Using a novel method based on ANN, the approximate solutions of FDEs were investigated in [28]. Neural network approach being a part of Artificial intelligence (AI) has become very popular due to its ability to do incredible tasks like finding the patterns in data which otherwise is very difficult. B. Shiri et al. in [18] proposed an adaptive gradient descent method based on NNM to minimize energy functions. For image processing, speech recognition, manufacturing virtual assistants like Alexa, training machines, this can perform more efficiently than a human being. ANN is also performing in those areas very efficiently, where human beings could not even think of it even one or two decades ago.

In present paper, we are putting forth a numerical method to solve the nonlinear time–space fractional PDEs. This method, consisting of Legendre polynomials, is motivated by the literary work [13], which employed the NNM to solve an integer order differential equation. Here we have made an effort to employ the method to solve a nonlinear FPDE, which is first of its kind.

The benefit of the studied method is its primary concept which is based on training the networks by using a small number of sample points on the solution area. It is noticed that the accuracy improves with more training. However it was very challenging to deal with the accuracy of the method with larger number of training set. The NNM is regarded as superior to the FEM, FDM or any other numerical method for solving the nonlinear PDEs in integer as well as fractional-order systems. This is because NNM can improve the calculation accuracy by encrypting the grid or by adding interpolating nodes. In contrast to the grid-based method, which can only find approximate solutions for grid points, NNM can obtain the approximate solutions for all of the points in the interval from a small sample of points FRADEs are in general solved using numerical methods. The scientific community is trying to develop and employ new and advanced techniques to achieve the more accurate solutions of FRADEs and also to reduce the effort in obtaining these solutions. Hence, state of the art methods developed in recent years such as NNM need to be tested for such problems. This led us to attempt the present study.

Here an endeavour is made to apply the NNM based on Legendre polynomials to investigate the following nonlinear time–space fractional order reaction-advection-diffusion equation (FRADE) given by

$$\frac{\partial^{\alpha} w(x,t)}{\partial t^{\alpha}} = \frac{\partial^{\beta} w(x,t)}{\partial x^{\beta}} + \nu w^{\eta}(x,t)\frac{\partial w(x,t)}{\partial x} + \lambda R(w), 0 < \alpha < 1, 1 < \beta < 2, \tag{1}$$

under the prescribed initial condition and boundary conditions as

$$w(x,0)=\Psi_1(x), 0\leq x\leq X, \tag{2}$$

$$w(0,t)=\Psi_2(t), 0\leq t\leq T, \tag{3}$$

$$\frac{\partial w(1,t)}{\partial x}=\Psi_3(t), 0\leq t\leq T, \tag{4}$$

where $\eta\in N$, the set of all natural numbers and ν is the advection coefficient.

The reaction term is considered here as $R(w)=w(x,t)(1-w^{\eta}(x,t))$ which fulfils the Lipschitz criterion:

$$|R(w_1)-R(w_2)|\leq L|w_1-w_2|, \qquad \forall w_1,w_2.$$

The remaining portion of this article is structured as follows. A few helpful definitions are introduced in Section 2, along with Legendre polynomials and their shifted counterparts. A thorough explanation of NNM is given in Section 3. For the aim of demonstrating the efficacy of the concerned method, two numerical examples are given in Section 4. Section 5 discusses the results of the present work. The last section, Section 6, concludes the article.

2. Preliminary information

2.1. Definition

In the Caputo sense, the partial derivative of fractional-order α of a function $w(x,t)$ with respect to x is defined as [17]

$$ {}_0^cD_x^{\alpha}w(x,t)=\begin{cases}\dfrac{1}{\Gamma(n-\alpha)}\int_0^x(x-\xi)^{n-\alpha-1}\dfrac{\partial^n w(\xi,t)}{\partial\xi^n}d\xi, & \text{if } n-1<\alpha<n,\\ \dfrac{\partial^n w(x,t)}{\partial x^n}, & \text{if } \alpha=n\in N.\end{cases} \tag{5}$$

2.2. Legendre polynomial

Legendre polynomials on $[-1,1]$ are defined by the following recursive relations given as [13]

$$l_0(x)=1,$$

$$l_1(x)=x,$$

$$l_{i+1}(x)=\frac{2i+1}{i+1}xl_i(x)-\frac{i}{i+1}l_{i-1}(x), i=1,2,3,\ldots;$$

Analytical form of Legendre polynomial is given as

$$L_i(x)=\sum_{k=0}^{i}\frac{(-1)^k(i+k)!}{(i-k)!}\frac{(1-x)^k}{2^k(k!)^2}, i=0,1,2,\ldots;$$

The shifted Legendre polynomial of degree i on the interval $[0,X]$ is therefore given by the explicit analytical form

$$L_{X,i}(x)=\sum_{s=0}^{i}\frac{(-1)^{i+s}(i+s)!}{(i-s)!}\frac{x^s}{X^s(s!)^2}, i=0,1,2,\ldots; \tag{6}$$

and the boundary values of $L_{X,i}(x)$ are

$$L_{X,i}(0)=(-1)^i,$$

$$L_{X,i}(X)=1.$$

3. Neural network method

In this section, we will talk about how NNM works to solve nonlinear FRADE. The qth test result is denoted by [13]:

$$w^q(x,t)=\sum_{i=o}^{M_x}\sum_{j=o}^{M_t}u^q_{i,j}L_{X,i}(x)L_{T,j}(t), \tag{7}$$

where $L_{X,i}(x)$ and $L_{T,j}(t)$ are the shifted Legendre polynomials stated in previous section. So, the function $w^q(x,t)$ defined in (7) is a continuous function on $[0,1]\times[0,1]$. The network is depicted in Fig. 3. The neural networks change the weights to reduce the loss function using the unknown weights $u^q_{i,j}$, $i\in\{0,1,\dots,M_x\}$ and $j\in\{0,1,\dots,M_t\}$ those are first allocated at random. Model (1) will be approximated by inserting the test solution $w^q(x,t)$ with unknown weights $u^q_{i,j}$ and iteratively training to change the unknown weights $u^q_{i,j}$.

The basis functions $L_{T,j}(t)$, and $L_{X,i}(x)$ have their time and spatial derivatives determined from the model (1). To begin, let us compute the time derivative. For convenience, let us consider

$$T(t)=[L_{T,0}(t),L_{T,1}(t),\dots,L_{T,M_t}(t)],$$

where $L_{T,j}(t)$ is defined in (6).

Case 1: If $\alpha=1$, we have

$$\frac{dT(t)}{dt}=\frac{d}{dt}[L_{T,0}(t),L_{T,1}(t),\dots,L_{T,M_t}(t)]=[0,L'_{T,1}(t),\dots,L'_{T,M_t}(t)], \tag{8}$$

where

$$L'_{T,j}(t)=\sum_{s=1}^{j}\frac{(-1)^{j+s}(j+s)!}{(j-s)!}\frac{x^{s-1}}{X^s(s!)(s-1)!},\ j=1,2,\dots,M_t. \tag{9}$$

Case 2: For $\alpha\in(0,1)$, we have

$${}^c_0D^\alpha_t T(t)=[0,L^\alpha_{T,1}(t),\dots,L^\alpha_{T,M_t}(t)], \tag{10}$$

where ${}^c_0D^\alpha_t$ is the Caputo derivative and it is defined as

$${}^c_0D^\alpha_t L_{T,j}(t)=L^\alpha_{T,j}(t)=\sum_{s=0}^{j}\frac{(-1)^{j+s}(j+s)!t^{s-\alpha}}{(j-s)!(s!)T^s\Gamma(s+1-\alpha)},\ j=1,2,\dots,M_t. \tag{11}$$

Now we will calculate spatial derivatives of

$$L(x)=[L_{X,0}(x),L_{X,1}(x),\dots,L_{X,M_x}(x)]^T.$$

For first order derivative of $L(x)$,

$$\frac{dL(x)}{dx}=[L'_{X,0}(x),L'_{X,1}(x),\dots,L'_{X,M_x}(x)]^T. \tag{12}$$

Similarly for second order derivative of $L(x)$,

$$\frac{d^2L(x)}{dx^2}=[L''_{X,0}(x),L''_{X,1}(x),\dots,L''_{X,M_x}(x)]^T, \tag{13}$$

and for $\beta\in(1,2)$, we get

$${}^c_0D^\beta_x L(x)=[L^\beta_{X,0}(x),L^\beta_{X,1}(x),\dots,L^\beta_{X,M_x}(x)], \tag{14}$$

where ${}^c_0D^\beta_x$ is Caputo derivative and it is defined by

$${}^c_0D^\beta_x L_{X,i}(x)=\sum_{s=0}^{i}\frac{(-1)^{i+s}(i+s)!x^{s-\beta}}{(i-s)!(s!)X^s\Gamma(s+1-\beta)}. \tag{15}$$

The parameter for the training set is chosen on uniform grid points as $x_m=\frac{(m-1)X}{(M_m-1)}$, $t_n=\frac{(n-1)T}{(M_n-1)}$, where $m\in\{1,2,\dots,M_m\}$ and $n\in\{1,2,\dots,M_n\}$.

Substituting (7) into the model (1) and using Eqs. (10), (12) and (14), the errors $er^q_{m,n}$ at sample points (x_m, t_n) for non-initial states $m \in \{2, 3, \dots, M_m - 1\}$ and $n \in \{2, 3, \dots, M_n\}$ are calculated as

$$\begin{aligned} er^q_{m,n} &= \sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L^{\alpha}_{T,j}(t_n) - \sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L^{\beta}_{X,i}(x_m) L_{T,j}(t_n) \\ &- \nu\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n)\Big)^{\eta}\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L'_{X,i}(x_m) L_{T,j}(t_n)\Big) - \lambda\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n)\Big) \\ &+ \lambda\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n)\Big)^{\eta+1}. \end{aligned} \tag{16}$$

While for initial condition (2), $n = 1$ and $m \in \{1, 2, \dots, M_m\}$ and we have

$$er^q_{m,1} = \sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_1) - \Psi_1(x_m). \tag{17}$$

For boundary condition (3), $m = 1$ and $n \in \{2, \dots, M_n\}$. Therefore,

$$er^q_{1,n} = \sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_1) L_{T,j}(t_n) - \Psi_2(t_n), \tag{18}$$

and for boundary condition (4), $m = M_m$ and $n \in \{2, \dots, M_n\}$ so that

$$er^q_{M_m,n} = \sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L'_{X,i}(x_{M_m}) L_{T,j}(t_n) - \Psi_3(t_n). \tag{19}$$

The qth error matrix is defined by $E^q = (er^q_{m,n})_{M_m \times M_n}$. The Frobenius matrix norm of E^q matrix is defined by

$$\|E^q\|_F^2 = \frac{1}{2}\sum_{m=1}^{M_m}\sum_{n=1}^{M_n}(er^q_{m,n})^2. \tag{20}$$

Next, the weight adjustment formula [13] is given by

$$u^{q+1}_{i,j} = u^q_{i,j} + \Delta u^q_{i,j}, \tag{21}$$

for $q = 0, 1, 2, \dots, N$, with

$$\Delta u^q_{i,j} = -\rho\frac{\partial \|E^q\|_F^2}{\partial u^q_{i,j}} = -\rho\sum_{m=1}^{M_m}\sum_{n=1}^{M_n} er^q_{m,n}\frac{\partial er^q_{m,n}}{\partial u^q_{i,j}}.$$

Case 1: When $m \in \{2, 3, \dots, M_m - 1\}$ and $n \in \{2, 3, \dots, M_n\}$, we get

$$\begin{aligned} \frac{\partial er^q_{m,n}}{\partial u^q_{i,j}} &= L_{X,i}(x_m) L^{\alpha}_{T,j}(t_n) - L^{\beta}_{X,i}(x_m) L_{T,j}(t_n) - \nu\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n)\Big)^{\eta}\big(L'_{X,i}(x_m) L_{T,j}(t_n)\big) \\ &- \nu\eta\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n)\Big)^{\eta-1}\big(L_{X,i}(x_m) L_{T,j}(t_n)\big)\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L'_{X,i}(x_m) L_{T,j}(t_n)\Big) \\ &- \lambda\big(L_{X,i}(x_m) L_{T,j}(t_n)\big) + \lambda(\eta+1)\Big(\sum_{i=0}^{M_x}\sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n)\Big)^{\eta}\big(L_{X,i}(x_m) L_{T,j}(t_n)\big). \end{aligned}$$

Case 2: When $n = 1$ and $m \in \{1, 2, \dots, M_m\}$, we obtain

$$\frac{\partial er^q_{m,1}}{\partial u^q_{i,j}} = L_{X,i}(x_m) L_{T,j}(t_1),$$

Case 3: When $m = 1$ and $n \in \{2, 3, \ldots, M_n\}$, we find

$$\frac{\partial er^q_{1,n}}{\partial u^q_{i,j}} = L_{X,i}(x_1) L_{T,j}(t_n).$$

Case 4: When $m = M_m$ and $n \in \{2, 3, \ldots, M_n\}$, we have

$$\frac{\partial er^q_{M_m,n}}{\partial u^q_{i,j}} = L'_{X,i}(x_{M_m}) L_{T,j}(t_n).$$

Hence,

$$\begin{aligned}
\Delta u^q_{i,j} = &-\rho \sum_{m=2}^{M_m-1} \sum_{n=2}^{M_n} er^q_{m,n} \Big\{ L_{X,i}(x_m) L^{\alpha}_{T,j}(t_n) - L^{\beta}_{X,i}(x_m) L_{T,j}(t_n) \\
&- \nu \big(\sum_{i=0}^{M_x} \sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n) \big)^{\eta} \big(L'_{X,i}(x_m) L_{T,j}(t_n) \big) \\
&- \nu\eta \big(\sum_{i=0}^{M_x} \sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n) \big)^{\eta-1} \big(L_{X,i}(x_m) L_{T,j}(t_n) \big) \big(\sum_{i=0}^{M_x} \sum_{j=0}^{M_t} u^q_{i,j} L'_{X,i}(x_m) L_{T,j}(t_n) \big) \\
&- \lambda \big(L_{X,i}(x_m) L_{T,j}(t_n) \big) + \lambda(\eta+1) \big(\sum_{i=0}^{M_x} \sum_{j=0}^{M_t} u^q_{i,j} L_{X,i}(x_m) L_{T,j}(t_n) \big)^{\eta} \big(L_{X,i}(x_m) L_{T,j}(t_n) \big) \Big\} \\
&- \rho \sum_{m=1}^{M_m} er^q_{m,1} \big(L_{X,i}(x_m) L_{T,j}(t_1) \big) - \rho \sum_{n=2}^{M_n} er^q_{1,n} \big(L_{X,i}(x_1) L_{T,j}(t_n) \big) - \rho \sum_{n=2}^{M_n} er^q_{M_m,n} \big(L'_{X,i}(x_{M_m}) L_{T,j}(t_n) \big).
\end{aligned}$$

Initial values of weights $u^0_{i,j}$ for $i = 0, 1, 2, \cdots, M_x$ and $j = 0, 1, 2, \cdots, M_t$ can be chosen at random. The maximum number of training allowed is N and the learning rate of the neural network is ρ.

Algorithm.

1. Construct sample points (x_m, t_n), where $m = 1, 2, \ldots, M_m$, and $n = 1, 2, \ldots, M_n$.
2. Generate the weights $u^0_{i,j}$ for $i = 0, 1, 2, \ldots, M_x$ and $j = 0, 1, 2, \ldots, M_t$ at random.
3. Calculate the Frobenius matrix's norm E^q, $q \leq N$.
4. Calculate the weights' increase $\Delta u^q_{i,j}$.
5. Adjust the weights' $u^q_{i,j}$ in accordance to the relation (21).
6. Take a rest if $\|E^q\|_F < \epsilon$ or training time $q < N$. Go to Step 3 if not.
7. Network test.

The theorem below discusses the learning rate ρ of NNM.

Theorem ([*13*]). *Assume that the parameter settings for the neural networks for model* (1) *are given by spatio-temporal neurons $M_x \times M_t$ and the number of sample points are $M_m \times M_n$. If the Lipschitz condition with parameter L is satisfied by the reaction term $R(w)$ and the constant associated with the Lipschitz parameter L and reaction term $R(w)$ is $M_{R,L} > 0$, then in order to guarantee that the error function reduces with practise, should satisfy*

$$0 < \rho < \frac{1 + \sqrt{M_m M_n}}{M^2_{R,L} M_m M_n M_x M_t}.$$

4. Numerical application

This section focuses on validating the suggested approach by employing it to solve two well-known standard test cases.

Example 1. *Let us consider the following FRADE [21] as*

$$\frac{\partial^{\alpha} w(x,t)}{\partial t^{\alpha}} = \frac{\partial^{\beta} w(x,t)}{\partial x^{\beta}} + w(x,t) \frac{\partial w(x,t)}{\partial x} + \lambda w(x,t)(1 - w(x,t)), \tag{22}$$

Table 1
Frobenius norm of error matrix for Example 1 at different number of training sets.

N	Error
10,000	2.607×10^{-4}
20,000	5.1217×10^{-5}
30,000	9.8995×10^{-6}
40,000	1.6785×10^{-6}

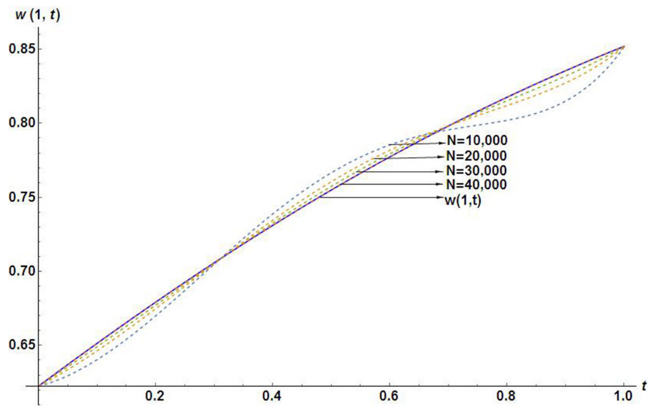

Fig. 1. Comparison of numerical solution of Example 1 for $N = 10{,}000$, $N = 20{,}000$, $N = 30{,}000$ and $N = 40{,}000$ with the exact solution at $x = 1$.

where $w(x, t)$ is a continuous function on $[0, 1] \times [0, 1]$ *with initial condition*

$$w(x, 0) = \frac{1}{2} + \frac{1}{2}\tanh\big(\frac{x}{4}\big), \tag{23}$$

and boundary conditions

$$w(0, t) = \frac{1}{2} + \frac{1}{2}\tanh\big(\frac{5t}{8}\big), \tag{24}$$

$$\frac{\partial w(1, t)}{\partial x} = \frac{1}{8}\,\mathrm{sech}^2\big(\frac{1}{4}\big(1 + \frac{5t}{2}\big)\big), \tag{25}$$

which has the exact solution [21] at $\alpha = 1$, $\beta = 2$ and $\lambda = 1$ as

$$w(x, t) = \frac{1}{2} + \frac{1}{2}\tanh\big(\frac{1}{4}\big(x + \frac{5t}{2}\big)\big). \tag{26}$$

This example represents a nonlinear model which is particularly challenging to solve. The goal of this example is to evaluate the precision of NNM when solving a nonlinear model. The following parameters are used to set up the model for spatial interval $X = 1$, time interval $T = 1$, $\eta = 1$, learning rate $\rho = 5 \times 10^{-5}$, $M_x = M_t = M_m = M_n = 5$. We can see from Table 1 that as number of training sets is increased, the error reduces and the accuracy of NNM improves. The comparison of the numerical solutions estimated by employing NNM with different number of training data points and exact solution is shown in Fig. 1. The graph perfectly demonstrates that increased number of training sets leads to enhanced numerical accuracy and also demonstrates that there is good agreement between the exact and numerical solutions.

Example 2. *Let us consider the following FRADE [23] given by*

$$\frac{\partial^\alpha w(x, t)}{\partial t^\alpha} = \frac{\partial^\beta w(x, t)}{\partial x^\beta} + w^2(x, t)\frac{\partial w(x, t)}{\partial x} + w(x, t)(1 - w^2(x, t)), \tag{27}$$

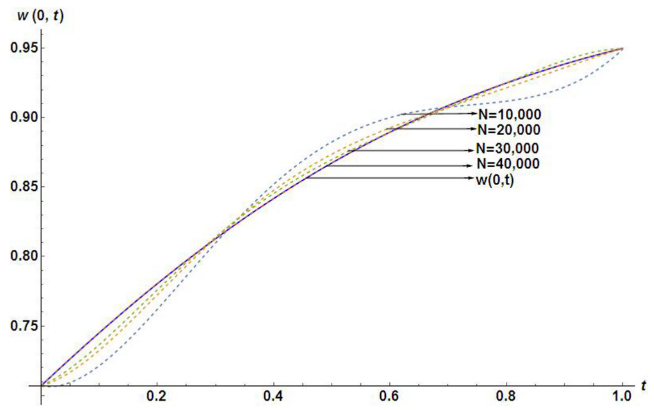

Fig. 2. Comparison of numerical solution of Example 2 for $N = 10{,}000$, $N = 20{,}000$, $N = 30{,}000$ and $N = 40{,}000$ with the exact solution at $x = 0$.

Table 2
Frobenius norm of error matrix for Example 2, at different number of training sets.

N	*Error*
10,000	4.0862×10^{-4}
20,000	5.8519×10^{-5}
30,000	8.57706×10^{-6}
40,000	1.1286×10^{-6}

where $w(x,t)$ is a function in $C[0,1] \times C[0,1]$ with initial condition

$$w(x,0) = \sqrt{\frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x}{3}\right)}, \tag{28}$$

and boundary conditions

$$w(0,t) = \sqrt{\frac{1}{2} + \frac{1}{2}\tanh\left(\frac{10t}{9}\right)}, \tag{29}$$

$$\frac{\partial w(1,t)}{\partial x} = \frac{\operatorname{sech}^2\left(\frac{1}{3}\left(1 + \frac{10t}{3}\right)\right)}{6\sqrt{2}\sqrt{1 + \tanh\left(\frac{1}{3}\left(1 + \frac{10t}{3}\right)\right)}}, \tag{30}$$

which has the exact solution [23] at $\alpha = 1$, $\beta = 2$ and $\lambda = 1$ as

$$w(x,t) = \sqrt{\frac{1}{2} + \frac{1}{2}\tanh\left(\frac{1}{3}\left(x + \frac{10t}{3}\right)\right)}. \tag{31}$$

As in previous example, this example also aims to demonstrate how NNM solves a nonlinear model accurately for $\eta = 2$ for the model (1). The model is configured using the parameters viz., spatial interval $X = 1$, time interval $T = 1$, and learning rate $\rho = 5 \times 10^{-5}$, $M_x = M_t = M_m = M_n = 5$. Table 2 demonstrates that as number of training sets is extended, the error decreases and NNM accuracy increases. Fig. 2 depicts the comparison of the exact solution and numerical results for different numbers of training sets. The figure clearly shows that more number of training sets result in higher numerical accuracy. The agreement between the numerical and exact results is also demonstrated in Fig. 2.

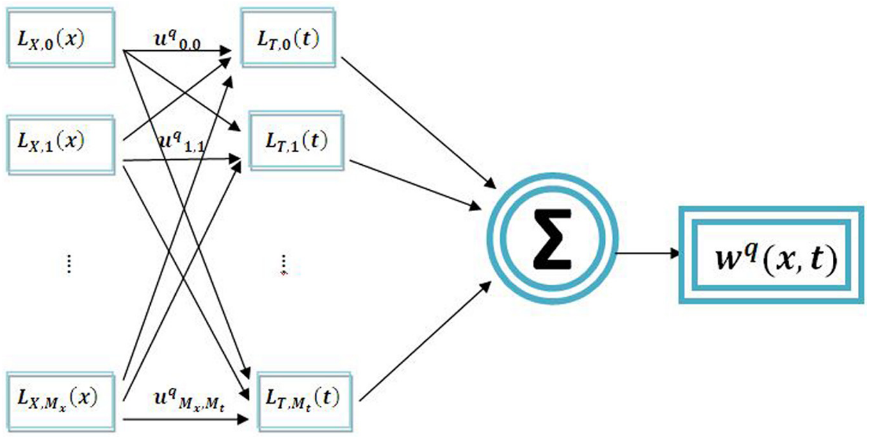

Fig. 3. Neural network structure.

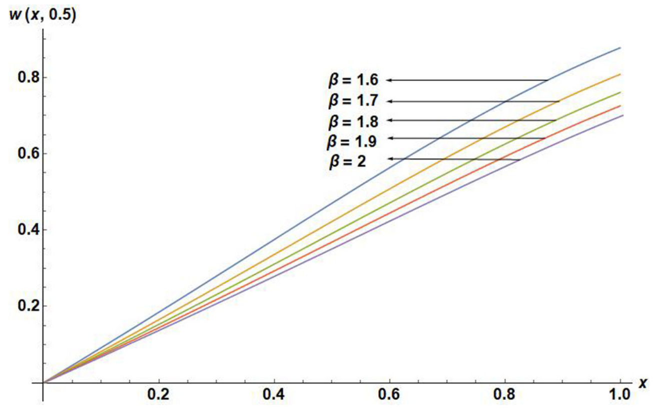

Fig. 4. Plots of concentration of the solute at fixed $t = 0.5$ for $\alpha = 0.6, \beta = 1.6, 1.7, 1.8, 1.9, 2$, $\lambda = -1$ and $\nu = 0.6$ where training set $N = 30{,}000$.

5. Application of NNM to solve the considered nonlinear FRADE

This section takes the following initial condition and boundary conditions into consideration as it attempts to solve the nonlinear FRADE given in (1).

$$w(x,0) = x^2, 0 \le x \le 1, \tag{32}$$

$$w(0,t) = 0, 0 \le t \le 1, \tag{33}$$

$$\frac{\partial w(1,t)}{\partial x} = e^{-t}, 0 \le t \le 1. \tag{34}$$

Our goal is to use the validated numerical method based on NNM to solve the model (1) with initial condition (32) and boundary conditions (33) and (34) for different particular cases. Fig. 4 shows the variations of solution for fixed $\alpha = 0.6$ and $\beta = 1.6, 1.7, 1.8, 1.9, 2$ and advection coefficient $\nu = 0.6$, at $\lambda = -1$ whereas Fig. 5 depicts the nature of solution for $\alpha = 1$ and $\beta = 1.6, 1.7, 1.8, 1.9, 2$ and $\nu = 0.6$ at $\lambda = -1$. The graph shows that the concentration of the solute drops as the order of the spatial derivative β moves from the fractional-order to the standard order. The solute travels a shorter distance in the soil column when β= 2 (standard order case) as compared to a fractional-order system, which is physically justified. Fig. 6 shows the solutions for fixed $\beta = 1.6$ and $\alpha = 0.6, 0.7, 0.8, 0.9, 1$ and advection coefficient $\nu = 0.6$ at $\lambda = -1$ and Fig. 7 depicts the nature of solution

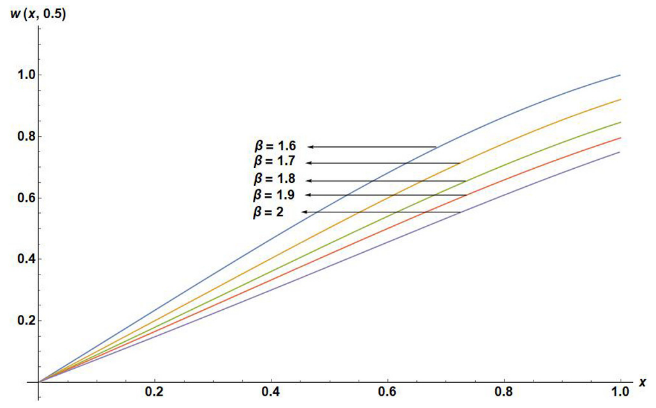

Fig. 5. Plots of concentration of the solute at fixed $t = 0.5$ for $\alpha = 1, \beta = 1.6, 1.7, 1.8, 1.9, 2,\ \lambda = -1$ and $\nu = 0.6$ where training set $N = 30{,}000$.

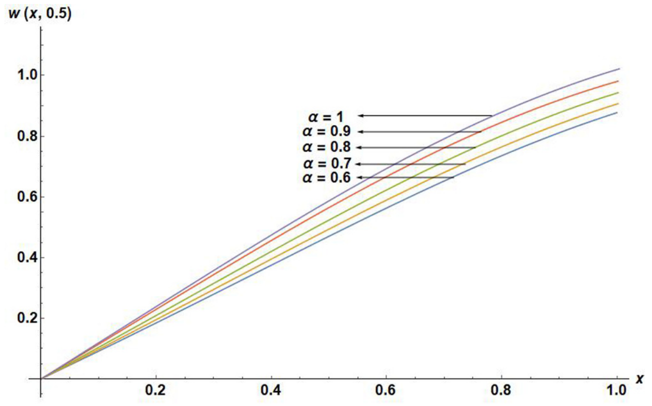

Fig. 6. Plots of concentration of the solute at fixed $t = 0.5$ for $\beta = 1.6, \alpha = 0.6, 0.7, 0.8, 0.9, 1,\ \lambda = -1$ and $\nu = 0.6$ where training set $N = 30{,}000$.

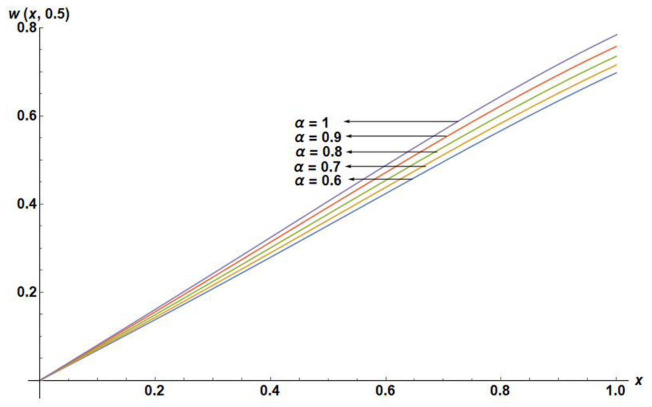

Fig. 7. Plots of concentration of the solute at fixed $t = 0.5$ for $\beta = 2, \alpha = 0.6, 0.7, 0.8, 0.9, 1,\ \lambda = -1$ and $\nu = 0.6$ where training set $N = 30{,}000$.

at fixed $\beta = 2$ and $\alpha = 0.6, 0.7, 0.8, 0.9, 1$ and $\nu = 0.6$ at $\lambda = -1$. These graphical representations demonstrate how the solute covers more length as the order of the time derivative α goes from fractional-order to standard order.

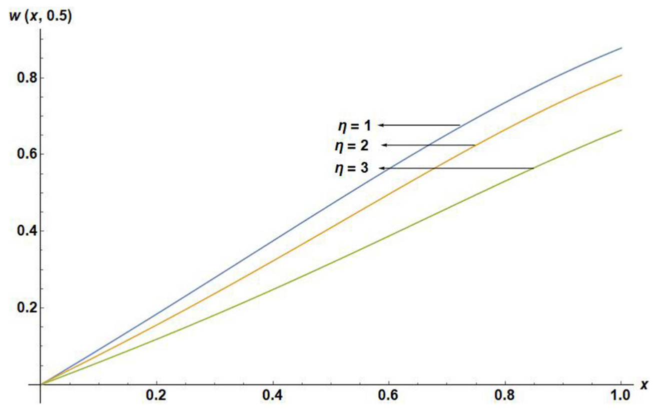

Fig. 8. Plots of concentration of the solute at fixed $t = 0.5$ for $\alpha = 0.6$, $\beta = 1.6$, $\eta = 1, 2, 3$ and $\nu = 0.6$ where training set $N = 30{,}000$.

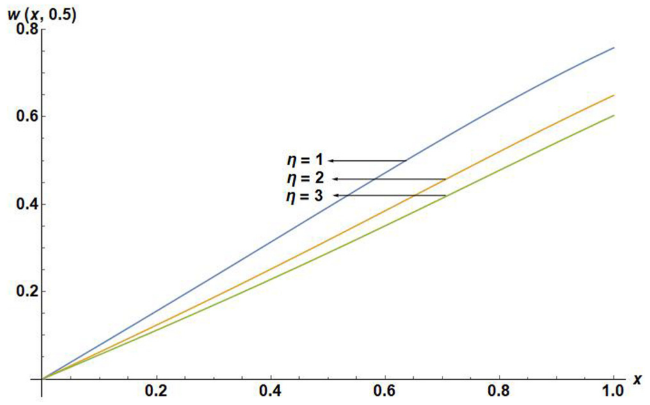

Fig. 9. Plots of concentration of the solute at fixed $t = 0.5$ for $\alpha = 1$, $\beta = 2$, $\eta = 1, 2, 3$ and $\nu = 0.6$ where training set $N = 30{,}000$.

Figs. 8 and 9 depict the solution profiles at fixed $\beta = 1.6$, $\alpha = 0.6$ and $\beta = 2$, $\alpha = 1$, respectively for $\lambda = -1$, $\nu = 0.6$ and $\eta = 1, 2, 3$. These graphical results show that the solute covers less distance in soil column with increase in the order of nonlinearity in the reaction term. Fig. 10 shows the numerical solution obtained at fixed $\beta = 2$ and $\alpha = 1$ at $\lambda = -1, 0, 1$ and $\nu = 0.6$. It is observed that for the fixed values of α and β, the concentration of the solute is less for the system with sink term ($\lambda = -1$), as compared to the system with conservative contaminant ($\lambda = 0$) and for the system with source term ($\lambda = 1$). This is physically justified that more damping will be found in the presence of sink term ($\lambda = -1$) as compared to the source term ($\lambda = 1$) as well as for the case of conservative system ($\lambda = 0$). It is noticed in all the graphs that the nature of the solution is similar for each case.

6. Conclusion

The nonlinear FRADE studied in this article is an attempt to determine its approximate solution under the specified initial and boundary conditions using NNM. This method includes training using a limited number of sample points in order to determine the weights of neurons. The trained neural network is then utilized for obtaining the solutions for the PDEs under unknown conditions. The applicability of the method is demonstrated by employing it to two test cases where exact solutions are known. Then the NNM is utilized to solve a FRADE. The results explains the concentration of solution profile decreases when the time derivative of the system approaches a standard order from the fractional order. The most crucial finding of the article is that, as the spatial order derivative decreases, the solute travels a shorter distance in the soil column for the standard order case ($\beta = 2$) as compared to a fractional-order system. As a result, the concentration of solute will cover a longer soil column length, which is evident from the visual representations of the data. The successful implementation of NNM for solving FRADE as done in the

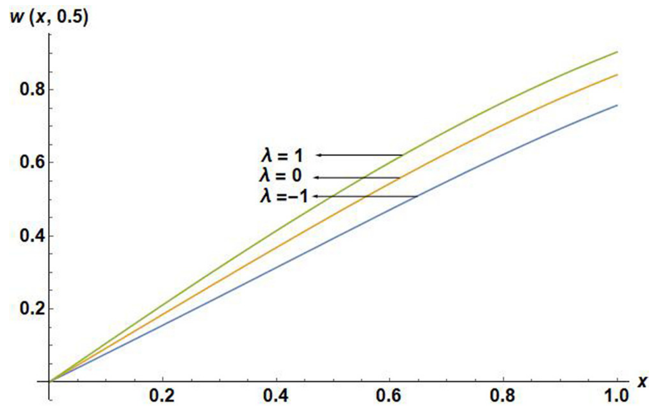

Fig. 10. Plots of concentration of the solute at fixed $t = 0.5$ for $\alpha = 1$, $\beta = 2$, $\lambda = 1, 0, -1$ and $\nu = 0.6$ where training set $N = 30{,}000$.

present work opens doors for future investigations on suitable basis functions for additional boundaries, like the Robin boundary or the Neumann boundary. After achieving the outstanding results while applying on the nonlinear time–space FRADE, the authors are confident to apply the proposed method to handle the nonlinear FRADE in two dimensional case and also solving many such nonlinear FPDEs under the prescribed Robin or Neumann boundary conditions having physical relevance. Moreover there is significant scope of upgrading the proposed method NNM to increase the accuracy, normalization and dropout of the method which will be taken care in near future during handling the nonlinear problems in fractional as well as integer order systems.

Acknowledgements

The authors would like to thank the reviewers for their constructive comments and suggestions to improve the quality of the revised article. The authors acknowledge the project grant provided by Board of Research in Nuclear Sciences (BRNS), BARC, Government of India (sanction no. 58/14/07/2022-BRNS/37041).

References

[1] A. Atangana, K. Owolabi, New numerical approach for fractional differential equations, Math. Model. Nat. Phenom. 13 (1) (2018) 3.
[2] S. Chen, S. Soradi-Zeid, H. Jahanshahi, R. Alcaraz, J.F. Gómez-Aguilar, S. Bekiros, Y. Chu, Optimal control of time-delay fractional equations via a joint application of radial basis functions and collocation method, Entropy 22 (11) (2020) 1213.
[3] S. Das, Analytical solution of a fractional diffusion equation by variational iteration method, Comput. Math. Appl. 57 (3) (2009) 483–487.
[4] S. Das, A note on fractional diffusion equations, Chaos Solitons Fractals 42 (4) (2009) 2074–2079.
[5] R. Hafez, M. Zaky, A. Hendy, A novel spectral Galerkin/Petrov–Galerkin algorithm for the multi-dimensional space–time fractional advection–diffusion–reaction equations with nonsmooth solutions, Math. Comput. Simulation 190 (2021) 678–690.
[6] 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.
[7] F. Kheyrinataj, A. Nazemi, Fractional Chebyshev functional link neural network-optimization method for solving delay fractional optimal control problems with Atangana-Baleanu derivative, Optim. Control Appl. Methods 41 (3) (2020) 808–832.
[8] M. Kumar, N. Yadav, Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: a survey, Comput. Math. Appl. 62 (10) (2011) 3796–3811.
[9] C. Li, M. Cai, Theory and Numerical Approximations of Fractional Integrals and Derivatives, SIAM, 2019.
[10] C. Li, F. Zeng, Numerical Methods for Fractional Calculus, vol. 24, CRC Press, 2015.
[11] V. Mishra, S. Das, H. Jafari, S.H. Ong, Study of fractional order Van der Pol equation, J. King Saud University-Science 28 (1) (2016) 55–60.
[12] M. Pakdaman, A. Ahmadian, S. Effati, S. Salahshour, D. Baleanu, Solving differential equations of fractional order using an optimization technique based on training artificial neural network, Appl. Math. Comput. 293 (2017) 81–95.
[13] H. Qu, X. Liu, X. Lu, M. ur Rahman, Z. She, Neural network method for solving nonlinear fractional advection-diffusion equation with spatiotemporal variable-order, Chaos Solitons Fractals 156 (2022) 111856.
[14] M.A.Z. Raja, J.A. Khan, I.M. Qureshi, Evolutionary computational intelligence in solving the fractional differential equations, in: Asian Conference on Intelligent Information and Database Systems, Springer, 2010, pp. 231–240.

[15] M.A.Z. Raja, R. Samar, M.A. Manzar, S.M. Shah, Design of unsupervised fractional neural network model optimized with interior point algorithm for solving Bagley–Torvik equation, Math. Comput. Simulation 132 (2017) 139–158.

[16] M. Saffarian, A. Mohebbi, Finite difference/spectral element method for one and two-dimensional Riesz space fractional advection–dispersion equations, Math. Comput. Simulation 193 (2022) 348–370.

[17] S. Shen, F. Liu, J. Chen, I. Turner, V. Anh, Numerical techniques for the variable order time fractional diffusion equation, Appl. Math. Comput. 218 (22) (2012) 10861–10870.

[18] B. Shiri, H. Kong, G. Wu, C. Luo, Adaptive learning neural network method for solving time–fractional diffusion equations, Neural Comput. 34 (4) (2022) 971–990.

[19] A. Singh, S. Das, Study and analysis of spatial-time nonlinear fractional-order reaction-advection-diffusion equation, J. Porous Media 22 (7) (2019).

[20] A. Singh, S. Das, S.H. Ong, Study and analysis of nonlinear (2+ 1)-dimensional solute transport equation in porous media, Math. Comput. Simulation 192 (2022) 491–500.

[21] A. Singh, K.D. Diwedi, S. Das, S.H. Ong, Study of one-dimensional space-time fractional-order Burgers-Fisher and Burgers-Huxley fluid models, Math. Methods Appl. Sci. 44 (3) (2021) 2455–2467.

[22] J.E. Solís-Pérez, J.A. Hernández, A. Parrales, J.F. Gómez-Aguilar, A. Huicochea, Artificial neural networks with conformable transfer function for improving the performance in thermal and environmental processes, Neural Netw. 152 (2022) 44–56.

[23] V. Tamboli, P. Tandel, Solution of the time-fractional generalized Burger–Fisher equation using the fractional reduced differential transform method, J. Ocean Eng. Sci. 7 (4) (2022) 399–407.

[24] M.H. Tber, A semi-Lagrangian mixed finite element method for advection–diffusion variational inequalities, Math. Comput. Simulation 204 (2023) 202–215.

[25] G. Tinoco-Guerrero, F.J. Domínguez-Mota, J. Tinoco-Ruiz, A study of the stability for a generalized finite-difference scheme applied to the advection–diffusion equation, Math. Comput. Simulation 176 (2020) 301–311.

[26] E. Viera-Martin, J.F. Gómez-Aguilar, J.E. Solís-Pérez, J.A. Hernández-Pérez, R.F. Escobar-Jiménez, Artificial neural networks: a practical review of applications involving fractional calculus, Eur. Phys. J. Spec. Top. (2022) 1–37.

[27] J. Wei, G. Wu, B. Liu, Z. Zhao, New semi-analytical solutions of the time-fractional Fokker–Planck equation by the neural network method, Optik 259 (2022) 168896.

[28] C.J. Zúñiga-Aguilar, H. Romero-Ugalde, J.F. Gómez-Aguilar, R.F. Escobar-Jiménez, M. Valtierra-Rodríguez, Solving fractional differential equations of variable-order involving operators with Mittag-Leffler kernel using artificial neural networks, Chaos Solitons Fractals 103 (2017) 382–403.