Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

Numerical Methods for Fractional Differential Equations Using Physics Informed Neural Networks

Sneha Agarwal, Lakshmi Narayan Mishra*

Department of Mathematics, School of Advanced Sciences, Vellore Institute of Technology, Vellore, Tamil Nadu, 632014, India
E-mail: lakshminarayanmishra04@gmail.com

Received: 15 July 2025; **Revised:** 8 September 2025; **Accepted:** 14 October 2025

Abstract: This paper investigates the numerical solution of fractional differential equations involving the Erdélyi-Kober fractional operator, which play a pivotal role in modeling memory-dependent and anomalous dynamic processes in applied mathematics, physics, and engineering, including anomalous diffusion, viscoelasticity, and heterogeneous heat conduction. We introduce a novel neural network-based framework that integrates Physics-Informed Neural Networks with the Theory of Functional Connections and employs an Extreme Learning Machine for efficient training. A new loss function is constructed using the L1 finite difference scheme in combination with Volterra integral equations, enabling accurate approximation of Erdélyi-Kober fractional derivatives while ensuring real-valued computations and minimal error. The proposed approach presents three main contributions over conventional method: (i) seamless integration of Theory of Functional Connections with Physics-Informed Neural Networks specifically for Erdélyi-Kober fractional derivatives, (ii) a novel loss function formulation tailored to the Erdélyi-Kober operator, and (iii) enhanced accuracy and computational efficiency. A neural network approach based on L1 is compared with the conventional L1 numerical scheme in order to solve fractional differential equations. The L1 discretization approach is used in a deep learning model in the proposed neural architecture to increase efficiency and accuracy. Using numerical analyses, we demonstrate that the L1 neural network technique achieves reliable convergence and exceeds the accuracy of the traditional L1 method. Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and coefficient of determination (R^2) are performance metrics that further support the model's effectiveness and demonstrate its resilience while handling fractional-order problems. The results show that the combination of neural networks and the L1 scheme provides a strong and accurate solution for solving complex fractional differential equations. Several illustrative examples are provided to demonstrate the efficacy, reliability, and practical relevance of the methodology. The results highlight the capability of the proposed framework to solve complex fractional differential equations effectively, offering a powerful tool for researchers and practitioners in applied mathematics, engineering, and related scientific disciplines.

Keywords: Erdélyi-Kober fractional derivative, L1 finite difference scheme, Neural Networks, Extreme Learning Machine (ELM), Theory of Functional Connections (TFC)

MSC: 26A33, 34K37, 68T07, 45D05

DOI: https://doi.org/10.37256/cm.7220267860

1. Introduction

Neural Networks (NNs) are highly accurate numerical methods that model Fractional Differential Equations (FDEs). The major goal of employing NN-based approximation is that, according to the universal approximation theorem, NNs may act as universal approximators, thereby providing a helpful tool for predicting the Differential Equation (DE). Moreover, the use of NNs assists the discretization process; therefore, it minimizes discretization-based physics errors. NNs-based numerical approaches are currently gaining success, with various researchers developing a number of extremely precise algorithms. In 2017, Pakdaman [1] studied the technique to approximate the solutions to FDEs by using artificial NNs, and this method was compared with the analytical solutions and several existing numerical techniques to show the efficiency of artificial NNs [2]. The standard Physics Informed Neural Networks (PINNs) technique fails to address the mechanics of the problem. The Theory of Functional Connections (TFC) offers an ideal foundation for the PINNs-based approach that fulfills the principles behind the structure analytically. The TFC is a recently proposed mathematical framework to perform functional interpolation, which can be considered a generalization of interpolation. In particular, this methodology analytically derives functionals, called Constrained Expressions (CE), to represent all functions satisfying a set of some given linear equality and constraints to transform constrained problems into unconstrained ones [3]. Later, Schiassi et al. [4] investigated the implementation of deep NNs as free functions for estimating partial differential equation solutions. They conducted evaluations on both linear and nonlinear cases.

Fractional differential and integral equations [5] involving the Riemann-Liouville fractional operator or the Caputo fractional operator [6] have received more and more attention; however, there is little work on differential equations and integral equations involving Erdélyi-Kober (EK) fractional operators. Since integral equations with EK fractional operators are usually applicable in the theory of radiative transfer and the kinetic theory of gases [7], the theory of neutron transport [8], and the traffic theory. Various fractional derivative models, where non-locality plays a crucial role, have been introduced in several domains, like engineering, physical mechanics and dynamical systems [9], signal and image processing [10], control theory [11], biology [12], environmental sciences, and materials [13]. Riemann-Liouville [14], Hadamard [15], and Caputo [16] are examples of fractional derivatives that are defined using fractional integrals. A large part of the research on fractional-order boundary problems is concerned with integral boundary conditions of the classical, Riemann-Liouville, or Hadamard types. Recent research on nonlinear optical models, such as the Lakshmanan-Porsezian-Daniel equation with parabolic law nonlinearity, has brought attention to the significance of obtaining accurate soliton solutions for understanding pulse propagation, dispersion management, and modulation instability in optical fibers [17]. These developments highlight the broader need for accurate numerical and analytical methods in complex differential equations. The nonlinear FDEs, including EK derivatives, provide significant analytical and numerical challenges because of their nonlocal character and the challenge of imposing initial conditions. A mesh-free, data-driven approach to overcoming these challenges is offered by physics-informed neural networks. By directly embedding the governing differential equations and initial/boundary conditions into a neural network's loss function, this technique enables precise and effective approximation of solutions. It is particularly important to apply these ideas to fractional differential equations since fractional models can better describe physical and technical systems by capturing memory effects, hereditary behavior, and anomalous transport. The use of fractional differential operators has many advantages. For instance, they are more effective and versatile tools for replicating real-world problems because of their non-locality attribute and great degree of freedom in their sequence of differentiation. Thus, many observable phenomena involving anomalous diffusion, fractal dynamics, and random walks are captured by derivatives of fractional orders. Fractional differential operators can display memory and inheritance effects due to the non-locality feature, which means that the subsequent states of a dynamical system depend on both its current state and its previous states [18]. In addition to the aforementioned criteria, the EK fractional integral operator is used in another sort of integral boundary condition. These operators are critical in solving single, dual, and triple integral equations [19] with kernels that contain special functions of mathematical physics. Furthermore, the applications of the EK fractional integrals have been discussed in [20]. The classical fractional calculus [21] is based on several definitions for the operators of integration and differentiation of arbitrary order. Among the various definitions of fractional differentiation, the Riemann-Liouville and Caputo fractional derivatives are widely used in the literature [22]. The most useful classical fractional integrals, however, seem to be the

EK fractional operators. These were introduced by Hanna [23], who studied their basic properties and emphasized their useful applications to generalized axially symmetric potential theory and other physical problems, such as electrostatics and elasticity [24]. These days, fractional calculus is thought to be among the most significant and well-known applied mathematics research areas also applied for variational iteration method, homotopy perturbation method, reduce difference transform method and many more [25, 26]. The authors [27] have recently used a non-singular kernel to develop a new fractional derivative. In research and engineering, these operators are frequently used to address a wide range of problems. Some recent uses for fractional-order operators include ecology [28], epidemiology [29], psychology [30], mechanics [31], optics [32], and chemical reactor theory [33].

The reason fractional calculus is so popular and has so many applications nowadays is because the differintegral is an operator that contains both integrals and integer-order derivatives as special instances [34]. For instance, the fractional integral can be utilized as a regression model parameter to better describe the accumulation of a quantity when the order of integration is unclear. This work is innovative because it embeds the EK derivative within the PINN framework, which allows extremely nonlinear fractional differential equations to be solved while maintaining consistent initial condition enforcement. A robust computational tool with potential applications in anomalous diffusion, viscoelastic materials, and complex fractional-order dynamic systems where traditional numerical methods may be computationally costly or insufficiently accurate is provided by the proposed approach, which combines the flexibility of neural networks with the generalized modeling capability of the EK operator.

The main aim of this work is to develop a neural network-based computational framework that integrates the theory of functional connections, physics-informed neural networks, and extreme learning machines to obtain accurate, efficient, and stable solutions of EK fractional differential equations. We suggest two approaches to solve EK fractional derivative. The first employs an L1-based method, whereas the second uses a Volterra integral-based approximation.

The article has been structured in the following manner: Section 2 summarizes the fundamental definitions and ideas of PINNs and Extreme Learning Machine (ELM). Section 3 outlines the major conclusions of discretizing the derivative and obtaining the Loss Function (LF).

This section comprises the following two subsections: subsection 3.1 explains the process of estimating fraction derivatives, and subsection 3.2 describes the multiple delay problem using the L1-based approximation and the Volterra Integral Equation (VIE)-based approximation. In Section 4, numerical results are given to solve the differential equations involving EK fractional derivatives. Finally, Section 5 presents the conclusions.

2. Preliminaries

This section provides the necessary definitions to specify the desired results.

Definition 1 [35] The right-hand side EK fractional integral of order δ is defined by

$$\mathscr{I}_{\rho}^{s,\,\delta}\zeta(y)=\frac{\rho y^{-\rho(\delta+s)}}{\Gamma(\delta)}\int_0^y(y^{\rho}-\mu^{\rho})^{\delta-1}\mu^{\rho s+\rho-1}\zeta(\mu)\,d\mu,$$

with $s\in\mathbb{R}$ and $\delta,\ \rho>0$, where $\mathbb{R}$ is real number.

Definition 2 [36] The right Caputo EK fractional derivative of order η of a function $\hbar$ and $s-1<\eta\leq s$ is defined as

$$({}_cD_{\tau}^{\gamma,\,\eta}\hbar)(x)=K_{\tau}^{\gamma+\eta,\,s-\eta}\left(\prod_{i=0}^{s-1}\left(\gamma+i-\frac{1}{\tau}t\frac{d}{dt}\right)\hbar\right)(x),$$

where $K_{\tau}^{\gamma,\,\eta}$ is right-hand side EK fractional integral.

Definition 3 [36] The left Caputo EK fractional derivative of order η of a function $\hbar$ and $s-1<\eta\leq s$ is defined as

$$({}_cD_{\tau}^{\gamma,\,\eta}\hbar)(x)=I_{\tau}^{\gamma+\eta,\,s-\eta}\left(\prod_{i=0}^{s-1}\left(1+\gamma+i+\frac{1}{\tau}t\frac{d}{dt}\right)\hbar\right)(x),$$

$I_{\tau}^{\gamma,\,\eta}$ is left-hand side EK fractional integral.

Definition 4 [37] Based on the concept of TFC, a function $\eta(x)$ with constraint t is to be transformed by CE as follows:

$$\eta_{CE}(x,\ \kappa(x))=\kappa(x)+\sum_{j=1}^{t}\mu_j(x,\ \kappa(x))\nu_j(x), \tag{1}$$

where $x\in[x_0,\,X],\ \kappa(x)\in C^t([x_0,\,X];\,\mathbb{R})$ with $\kappa(x_0)\neq\infty$ is any freely chosen function such as the legendre series or neural network, and h_i represents the collection of all functions with linear independence, where $h_i\neq 0$, $\mu_j(x)$ is constructed in such a way that the CE $\eta_{CE}(x,\ \kappa(x))$ fulfills the t constraints for the unidentified value $\eta(x)$. Now, let us consider a function with the following constraints

$${}_0D_{x_1}^{\sigma_1,\,\rho_1}\eta(x)=\eta_1^{\sigma_1,\,\rho_1},\ {}_0D_{x_2}^{\sigma_2,\,\rho_2}\eta(x)=\eta_2^{\sigma_2,\,\rho_2},\ \cdots,\ {}_0D_{x_t}^{\sigma_t,\,\rho_t}\eta(x)=\eta_t^{\sigma_n,\,\rho_n}. \tag{2}$$

The constrained equation for the function $\eta(x)$ can be generated as shown in Eq. (1)
The values of $\mu_i,\ i=1,\ \cdots,\ t$ are computed using Eq. (1) and (2) as

$$\begin{pmatrix}\mu_1\\ \mu_2\\ \mu_3\\ \vdots\\ \mu_t\end{pmatrix}=\begin{pmatrix}{}_0^cD_{x_1}^{\sigma_1,\,\rho_1}\nu_1(x) & {}_0^cD_{x_1}^{\sigma_1,\,\rho_1}\nu_2(x) & \cdots & {}_0^cD_{x_1}^{\sigma_1,\,\rho_1}\nu_t(x)\\ {}_0^cD_{x_2}^{\sigma_2,\,\rho_2}\nu_1(x) & {}_0^cD_{x_2}^{\sigma_2,\,\rho_2}\nu_2(x) & \cdots & {}_0^cD_{x_2}^{\sigma_2,\,\rho_2}\nu_t(x)\\ \vdots & \vdots & & \vdots\\ {}_0^cD_{x_t}^{\sigma_t,\,\rho_t}\nu_1(x) & {}_0^cD_{x_t}^{\sigma_t,\,\rho_t}\nu_2(x) & \cdots & {}_0^cD_{x_t}^{\sigma_t,\,\rho_t}\nu_t(x)\end{pmatrix}^{-1}\times\begin{pmatrix}\eta_1^{\sigma_1,\,\rho_1}-{}_0^cD_{x_1}^{\sigma_1,\,\rho_1}\kappa(x)\\ \eta_2^{\sigma_2,\,\rho_2}-{}_0^cD_{x_2}^{\sigma_2,\,\rho_2}\kappa(x)\\ \vdots\\ \eta_t^{\sigma_t,\,\rho_t}-{}_0^cD_{x_t}^{\sigma_t,\,\rho_t}\kappa(x)\end{pmatrix}.$$

It should be emphasized that the constraints for a EK DE are always realistically relevent and hence $\sigma_j=1,\ \rho_j=1,$ $\forall j=\{1,\,2,\,\cdots,\,t\}$.

2.1 *Physics informed neural networks*

This section presents a framework, called Physics-Informed Neural Networks (PINNs), for solving nonlinear Differential Equations (DEs). Unlike traditional numerical solvers, which may require accurate discretization or specific techniques to handle significant nonlinearities, PINNs leverage the structure of the DE itself by integrating the governing equation and its constraints into the neural network's training process.

The basic concept is to develop a Loss Function (LF) that penalizes deviations from both the differential equation and its initial/boundary conditions. This trains the network to fit data and follow the physical rules described by the DE. Which helps to give a solution that follows the basic principles of the differential equation.

Let us take the differential equation as follows:

$$\frac{d\eta}{dx} = F(x,\ \eta(x)), \tag{3}$$

using the initial condition as

$$\eta(0) = \eta_0,$$

where $x \in [c,\ d]$. To use physics informed neural network, the domain x must first be reduced into t parts as $x_1 = c,\ x_2,\ \cdots,\ x_{t-1},\ x_t = d$. Then, NN is constructed to approximate the solution of a differential equation.

In [38], considering a single NN with three layers: input, hidden, and output layers, the LF is created using the NN's output $O(.)$ in the following way:

$$L = L_{EQ} + L_{IC},$$

where,

$$L_{EQ} = \sum_{i=1}^{t}\left[\left(\frac{dO(x)}{dx}\right)_{x=x_i} - F(x_i,\ O(x_i))\right]^2,$$

$$L_{IC} = (O(0) - \eta_0)^2.$$

A minimization problem is now designed to minimize L in terms of NN weights and constraints. This can be accomplished by using any optimized strategy that generates the NN's output, which is the differential equation's solution. The LF is expanded to incorporate a boundary loss if boundary constraints are stated at both ends, for instance, $\eta(c) = \eta_c$ and $\eta(d) = \eta_d$.

$$L_{BC} = (O(c) - \eta_c)^2 + (O(d) - \eta_d)^2,$$

obtaining the total loss function:

$$L_{total} = L_{EQ} + L_{IC} + L_{BC}.$$

PINNs transform the nonlinear DE problems into a function space restricted optimization problem, where the neural network functions as a universal approximator under severe physical constraints. This approach not only yields accurate and stable solutions for highly nonlinear problems, but it also readily generalizes to PDEs, higher-order equations, and inverse scenarios where training simultaneously learns unknown parameters in $F(x,\ \eta)$.

2.2 *Extreme learning machine*

The Extreme Learning Machine (ELM) [39] is designed to train Single Hidden-Layer Feedforward Neural Networks (SLFNs). According to neural network learning algorithm, it enhance the training speed and over fitting problem. This approach uses minimal computational resources to train NNs efficiently and quickly. Now, to train a NN, let inputs and targets are I_n and S_n, respectively, where $n \in \mathbb{N}$ and $\mathbb{N}$ is natural number. If the input I_n provides the output O_n, then the loss function of NN is given as

$$L = \sum_{i=1}^{j} (O_i - I_i).$$

Let a single-layer NN is given as follows:

$$O_i = \sum_{j=1}^{s} \gamma_j (\varsigma(W_{ji} S_i + B_j)),$$

γ_j represents the weight between the output layer and the j^{th} neuron in the hidden layer, the bias term applied to the j^{th} hidden node is expressed by B_j, and W_{ji} represented the weight between the input and the j^{th} neuron in the hidden layer.

Vector B and matrix W both possess fixed values and the weights of the hidden and output layers are adjusted to train the NNs. So, training these NNs becomes the training of a regression model [40] using the Least Squares (LS) formula.

$$M\gamma = I,$$

where,

$$M = \begin{pmatrix} \rho(W_{11}S_1 + B_1) & \rho(W_{21}S_1 + B_2) & \cdots & \rho(W_{s1}S_1 + B_s) \\ \rho(W_{12}S_2 + B_1) & \rho(W_{22}S_2 + B_2) & \cdots & \rho(W_{s2}S_2 + B_s) \\ \vdots & \vdots & & \vdots \\ \rho(W_{1j}S_j + B_1) & \rho(W_{2j}S_j + B_2) & \cdots & \rho(W_{sj}S_j + B_s) \end{pmatrix}, \ \gamma = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_s \end{pmatrix}, \ I = \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_j \end{pmatrix}.$$

The Moore-Penrose inverse idea can be used in the following ways to invert a linear system:

$$\gamma = M^T (M^T M)^{-1} I.$$

Assuming the system is nonlinear, we can determine the minimal value of γ using either the Newton's technique or nonlinear LS technique. In the nonlinearity case, the desired result for minimization is expressed as:

$$L(S,\ \gamma) = \begin{pmatrix} L_1 \\ L_2 \\ \vdots \\ L_j \end{pmatrix},$$

where each L_n is nonlinear.

Our goal is to find γ in order to reduce L. Now, to achieve this, we build objective-specific Jacobian matrices starting with an estimate of γ_0 and modify the estimated value of the unknown weights γ_l at the $(l+1)^{th}$ iteration using an iteratively changed vector $\Delta\gamma_l$, using the modified formula as

$$\mathscr{J}(\gamma_l)\Delta\gamma_l = -L_l,$$

where, the jacobian matrix at $\gamma = \gamma_l$ is denoted by $\mathscr{J}(\gamma_l)$, and the modified value for γ_l in the $l+1$ iteration can be written as

$$\gamma_{l+1} = \gamma_l + \Delta\gamma_l,$$

where $\Delta\gamma_l$ can be expressed in the following way:

$$\Delta\gamma_l = -(\mathscr{J}^T(\gamma_l)\mathscr{J}(\gamma_l))^{-1}\mathscr{J}^T(\gamma_l)L(\gamma_l).$$

This iteration method continues as long as the value of $L(\gamma_l) < \varepsilon$ for a predetermined tolerance ε.

3. Illustration of the method

The salient outcomes of the LF design for NN training are presented in this section.

3.1 *Methods for approximating fractional derivatives*

The following subsection discusses two distinct approximation methods applied using the NN to create the LF.

3.1.1 *Approximation of the fractional derivative using L1*

This portion discusses the L1-based discretization of the EK fractional derivative. We apply this method to extend the L1 discritization of the EK fractional derivative. The L1-based discretization is primarily designed to lower the computing expenses for developing the NNs. Now, let's evaluate the most significant problem as follows:

$$({}_cD_\tau^{\gamma,\ \eta}\hbar)(x) = \digamma(x,\ \hbar(x)), \tag{4}$$

where $1 < \eta < 2$, $x \in [c,\ d]$. Divide the interval into w points as $x_1,\ \cdots,\ x_w$, with a step size of dx. Let us consider the left-hand side of Eq. (4)

$$({}_cD_\tau^{\gamma,\,\eta}\hbar)(x) = I_\tau^{\gamma+\eta,\,s-\eta}\left(\prod_{i=0}^{s-1}(1+\gamma+i+\frac{1}{\tau}t\frac{d}{dt})\hbar\right)(x).$$

Now, at $x = x_w,\ s = 1$, and $\gamma = -1$, we have

$$({}_cD_\tau^{-1,\,\eta}\hbar)(x_k) = \frac{\tau}{\Gamma(1-\eta)}\int_c^{x_w}(x_w^\tau - t^\tau)^{-\eta}t^{\tau\eta-1}\left(\frac{t}{\tau}\frac{d}{dt}\right)\hbar(t)dt.$$

Now, substituting $t^\tau = u$ and fixing $\tau\eta = 0$, we get

$$({}_cD_\tau^{-1,\,\eta}\hbar)(x_k) = \frac{1}{\Gamma(1-\eta)}\int_{c^\tau}^{x_w^\tau}(x_w^\tau - u)^{-\eta}\hbar'(u^{1/\tau})\frac{du}{\tau(u^{1/\tau})^{\tau-1}},$$

$$= \frac{1}{\tau\Gamma(1-\eta)}\sum_{j=1}^{w-1}\left[\int_{x_j^\tau}^{x_{j+1}^\tau}\left((x_w^\tau - u)^{-\eta}(u^{1/\tau})^{1-\tau}du\right)\left(\frac{\hbar(x_{j+1})-\hbar(x_j)}{dx}\right)\right].$$

Now, we can approximate the integral value using spline interpolation as follows:

$$({}_cD_\tau^{-1,\,\eta}\hbar)(x_k)$$

$$=\frac{1}{\tau\Gamma(1-\eta)}\sum_{j=1}^{w-1}\left\{\begin{array}{l}\int_{x_j^\tau}^{x_{j+1}^\tau}\left[(x_w^\tau-u)^{-\eta}\dfrac{(u-x_{j+1}^\tau)}{x_j^\tau-x_{j+1}^\tau}(x_j)^{1-\tau}+\dfrac{u-x_j^\tau}{x_{j+1}^\tau-x_j^\tau}\right.\\ \left.\times(x_w^\tau-u)^{-\eta}(x_{j+1})^{1-\eta}\right]du\left(\dfrac{\hbar(x_{j+1})-\hbar(x_j)}{dx}\right)\end{array}\right\}.$$

$$= \frac{1}{\tau\Gamma(3-\eta)} \sum_{j=1}^{w-1} \left\{ \left(\frac{\hbar(x_j) - \hbar(x_{j+1})}{dx} \right) \left[\begin{array}{l} \left(\frac{(x_j)^{1-\tau}}{x_{j+1}^{\tau} - x_j^{\tau}} \right) \begin{pmatrix} (2-\eta)(x_w^{\tau} - x_{j+1}^{\tau}) \\ \times \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} \right. \\ \left. -(x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ -\left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} \right. \\ \left. -(x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{pmatrix} \\ + \left(\frac{(x_{j+1})^{1-\tau}}{x_j^{\tau} - x_{j+1}^{\tau}} \right) \begin{pmatrix} (2-\eta)(x_w^{\tau} - x_j^{\tau}) \\ \times \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} \right. \\ \left. -(x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ -\left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} \right. \\ \left. -(x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{pmatrix} \end{array} \right] \right\}. \tag{5}$$

Here, at $\eta = 2$, the derivative becomes zero, leading to the failure of the approximation. This result is applicable for $1 < \eta < 2$. Therefore, to address this issue, we employ the Volterra integral equation at $\eta = 2$.

3.1.2 *Approximation of the fractional derivative using the Volterra integral equation*

In this subsection, we will look at another way of discretizing the equations. The FDE Eq. (4) can be transformed into a VIE in the following way:

$$\hbar(x_w) = v(x_w) + \frac{\tau}{\Gamma\eta} x^{-\tau(\gamma+\eta)} \int_0^x (x^{\tau} - p^{\tau})^{\eta-1} p^{\tau(\gamma+1)-1} F(p,\, \hbar(p)) dp.$$

Let $p^{\tau} = \zeta$, then we get the following equation

$$\hbar(x_w) = v(x_0) + \frac{\tau}{\Gamma\eta} x^{-\tau(\gamma+\eta)} \int_0^{x_w} (x_w^{\tau} - \zeta)^{\eta-1} (\zeta^{1/\tau})^{\tau(\gamma+1)-1} F\left(\zeta^{1/\tau},\, \hbar(\zeta^{1/\tau}) \right) \frac{d\zeta}{\tau(\zeta^{1/\tau})^{\tau-1}}.$$

$$\hbar(x_w) = v(x_0) + \frac{x^{-\tau(\gamma+\eta)}}{\Gamma\eta} \int_0^{x_w} (x_w^{\tau} - \zeta)^{\eta-1} (\zeta^{1/\tau})^{\tau\gamma} F\left(\zeta^{1/\tau},\, \hbar(\zeta)^{1/\tau} \right) d\zeta.$$

Now divide the interval $[0,\, x]$ into w points $x_1,\, x_2,\, \cdots,\, x_w$, and the above equation becomes

$$\hbar(x_w) = v(x_0) + \frac{x^{-\tau(\gamma+\eta)}}{\Gamma\eta} \sum_{j=1}^{w-1} \int_{x_j^\tau}^{x_{j+1}^\tau} (x_w^\tau - \zeta)^{\eta-1} (\zeta^{1/\tau})^{\tau\gamma} F\left(\zeta^{1/\tau},\, \hbar(\zeta^{1/\tau})\right) d\zeta.$$

In order to approximate the function $F(x,\, \hbar(x))$, we used linear spline-based interpolation, resulting in the following:

$$\hbar(x_w) = v(x_0) + \frac{x^{-\tau(\gamma+\eta)}}{\Gamma\eta} \sum_{j=1}^{w-1} \left[\begin{array}{l} \dfrac{F(x_j,\, \hbar(x_j))}{(x_j^\tau - x_{j+1}^\tau)} \left(\displaystyle\int_{x_j^\tau}^{x_{j+1}^\tau} (x_w^\tau - \zeta)^{\eta-1} (x_j)^{\tau\gamma} (\zeta - x_{j+1}^\tau) d\zeta \right) \\ \\ + \dfrac{F(x_{j+1},\, \hbar(x_{j+1}))}{x_{j+1}^\tau - x_j^\tau} \left(\displaystyle\int_{x_j^\tau}^{x_{j+1}^\tau} (x_w^\tau - \zeta)^{\eta-1} (x_{j+1})^{\tau\gamma} (\zeta - x_j^\tau) d\zeta \right) \end{array} \right].$$

On solving further, we get the following result:

$$\hbar(x_w) = v(x_0) + \frac{x^{-\tau(\gamma+\eta)}}{\Gamma\eta} \sum_{j=1}^{w-1} \left\{ \begin{array}{l} \left(\dfrac{F(x_j,\, \hbar(x_j))}{x_j^\tau - x_{j+1}^\tau} \right) (x_j)^{\tau\gamma} \left[\begin{array}{l} \dfrac{x_{j+1}^\tau - x_w^\tau}{\eta} \left((x_w^\tau - x_{j+1}^\tau)^\eta - (x_w^\tau - x_j^\tau)^\eta \right) \\ -\dfrac{1}{\eta(\eta+1)} \left((x_w^\tau - x_{j+1}^\tau)^{\eta+1} - (x_w^\tau - x_j^\tau)^{\eta+1} \right) \end{array} \right] \\ \\ + \left(\dfrac{F(x_{j+1},\, \hbar(x_{j+1}))}{x_{j+1}^\tau - x_j^\tau} \right) (x_{j+1})^{\tau\gamma} \left[\begin{array}{l} \dfrac{x_j^\tau - x_w^\tau}{\eta} \left((x_w^\tau - x_{j+1}^\tau)^\eta - (x_w^\tau - x_j^\tau)^\eta \right) \\ -\dfrac{1}{\eta(\eta+1)} \left((x_w^\tau - x_{j+1}^\tau)^{\eta+1} - (x_w^\tau - x_j^\tau)^{\eta+1} \right) \end{array} \right] \end{array} \right\}. \tag{6}$$

This method approximates the EK fractional derivative for $1 < \eta \leq 2$.

3.2 *Multiple delay problem*

This section introduces a novel NN-based approach to solve the Erdélyi-Kober FDE that uses multiple delays. We offer two neural network-based approaches for approximating the solutions of DEs using multiple delays. Consider the Caputo EK fractional derivative with multiple delay given as

$${}_cD_\tau^{\gamma,\,\eta}\hbar(x) = F\left(x,\, \hbar(x),\, \hbar(x-\xi_1),\, \hbar(x-\xi_2),\, \cdots,\, \hbar(x-\xi_j)\right), \tag{7}$$

where $\xi_1,\, \xi_2,\, \cdots,\, \xi_j$ are the delays, the derivative's order is represented by η and $1 < \eta \leq 2$.

3.2.1 *L1 based approximation*

We employ the NN and L1-based approximation of Erdélyi-Kober fractional derivative given in Section 3.1.1 for solving DE (5) using the L1-method. Firstly, we divide the domain into w parts, denoted by $x_1 \cdots x_w$. After discretizing the domain, the EK fractional derivative is estimated using the L1-based approximation, as follows:

$$\frac{1}{\tau\Gamma(3-\eta)}\sum_{j=1}^{w-1}\left\{\left(\frac{\hbar(x_j)-\hbar(x_{j+1})}{dx}\right)\left[\begin{array}{c}\left(\frac{(x_j)^{1-\tau}}{x_{j+1}^{\tau}-x_j^{\tau}}\right)\left(\begin{array}{c}(2-\eta)(x_w^{\tau}-x_{j+1}^{\tau})\times\\ \left((x_w^{\tau}-x_{j+1}^{\tau})^{1-\eta}-(x_w^{\tau}-x_j^{\tau})^{1-\eta}\right)\\ -\left((x_w^{\tau}-x_{j+1}^{\tau})^{2-\eta}-(x_w^{\tau}-x_j^{\tau})^{2-\eta}\right)\end{array}\right)\\ +\frac{(x_{j+1})^{1-\tau}}{x_j^{\tau}-x_{j+1}^{\tau}}\left(\begin{array}{c}(2-\eta)(x_w^{\tau}-x_j^{\tau})\times\\ \left((x_w^{\tau}-x_{j+1}^{\tau})^{1-\eta}-(x_w^{\tau}-x_j^{\tau})^{1-\eta}\right)\\ -\left((x_w^{\tau}-x_{j+1}^{\tau})^{2-\eta}-(x_w^{\tau}-x_j^{\tau})^{2-\eta}\right)\end{array}\right)\end{array}\right]\right\}$$

$$=F\Big(x_i,\ \hbar(x_i),\ \hbar(x_i-\xi_1),\ \hbar(x_i-\xi_2)\cdots\hbar(x_i-\xi_n)\Big).$$

After that, the LF is created for solving the DE. The loss at the input value $x_i \in (x_1,\ x_2,\ \cdots,\ x_w)$ can be calculated as

$$L_i=\left[\begin{array}{c}\frac{1}{\tau\Gamma(3-\eta)}\sum_{j=1}^{w-1}\left\{\left(\frac{\hbar(x_j)-\hbar(x_{j+1})}{dx}\right)\left[\begin{array}{c}\left(\frac{(x_j)^{1-\tau}}{x_{j+1}^{\tau}-x_j^{\tau}}\right)\left(\begin{array}{c}(2-\eta)(x_w^{\tau}-x_{j+1}^{\tau})\times\\ \left((x_w^{\tau}-x_{j+1}^{\tau})^{1-\eta}-(x_w^{\tau}-x_j^{\tau})^{1-\eta}\right)\\ -\left((x_w^{\tau}-x_{j+1}^{\tau})^{2-\eta}-(x_w^{\tau}-x_j^{\tau})^{2-\eta}\right)\end{array}\right)\\ +\frac{(x_{j+1})^{1-\tau}}{x_j^{\tau}-x_{j+1}^{\tau}}\left(\begin{array}{c}(2-\eta)(x_w^{\tau}-x_j^{\tau})\times\\ \left((x_w^{\tau}-x_{j+1}^{\tau})^{1-\eta}-(x_w^{\tau}-x_j^{\tau})^{1-\eta}\right)\\ -\left((x_w^{\tau}-x_{j+1}^{\tau})^{2-\eta}-(x_w^{\tau}-x_j^{\tau})^{2-\eta}\right)\end{array}\right)\end{array}\right]\right\}\\ -F\Big(x_i,\ \hbar(x_i),\ \hbar(x_i-\xi_1),\ \hbar(x_i-\xi_2)\cdots\hbar(x_i-\xi_n)\Big)\end{array}\right]^2.$$

We now create the constrained equation for the NN using TFC-based interpolation, as described in Definition 2.The constrained equation of the function $\hbar(x)$ is generated as follows:

$$\hbar_{CE}(x,\ \mathscr{G}(x))=\mathscr{G}(x)+\rho(t).$$

The free function $\mathscr{G}(x)$ is represented as a NN with hidden layer 1 and neurons 10, with the reference function $h(t)$ set to 1. The neural network's activation function is a sigmoid function, given by

$$\sigma(t)=\frac{1}{1+e^{-t}}.$$

To obtain the loss at t_i, we are now incorporating the NN's output into the LF, as illustrated below:

$$L_i = \left[\frac{1}{\tau\Gamma(3-\eta)} \sum_{j=1}^{w-1} \left\{ \left(\frac{\hbar_{CE}(x_j) - \hbar_{CE}(x_{j+1})}{dx} \right) \left[\begin{array}{c} \left(\frac{(x_j)^{1-\tau}}{x_{j+1}^{\tau} - x_j^{\tau}} \right) \left(\begin{array}{c} (2-\eta)(x_w^{\tau} - x_{j+1}^{\tau}) \times \\ \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} - (x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ - \left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} - (x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{array} \right) \\ + \frac{(x_{j+1})^{1-\tau}}{x_j^{\tau} - x_{j+1}^{\tau}} \left(\begin{array}{c} (2-\eta)(x_w^{\tau} - x_j^{\tau}) \times \\ \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} - (x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ - \left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} - (x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{array} \right) \end{array} \right] \right\} \right]^2 .$$

$$- F\left(x_i,\ \hbar_{CE}(x_i),\ \hbar_{CE}(x_i - \xi_1),\ \hbar_{CE}(x_i - \xi_2) \cdots \hbar_{CE}(x_i - \xi_n) \right)$$

The entire loss of the equation associated with the NNs is displayed as

$$L_i = \sum_{i=1}^{n} \left[\frac{1}{\tau\Gamma(3-\eta)} \sum_{j=1}^{w-1} \left\{ \left(\frac{\hbar_{CE}(x_j) - \hbar_{CE}(x_{j+1})}{dx} \right) \left[\begin{array}{c} \left(\frac{(x_j)^{1-\tau}}{x_{j+1}^{\tau} - x_j^{\tau}} \right) \left(\begin{array}{c} (2-\eta)(x_w^{\tau} - x_{j+1}^{\tau}) \times \\ \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} - (x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ - \left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} - (x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{array} \right) \\ + \frac{(x_{j+1})^{1-\tau}}{x_j^{\tau} - x_{j+1}^{\tau}} \left(\begin{array}{c} (2-\eta)(x_w^{\tau} - x_j^{\tau}) \times \\ \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} - (x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ - \left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} - (x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{array} \right) \end{array} \right] \right\} \right]^2 .$$

$$- F\left(x_i,\ \hbar_{CE}(x_i),\ \hbar_{CE}(x_i - \xi_1),\ \hbar_{CE}(x_i - \xi_2) \cdots \hbar_{CE}(x_i - \xi_n) \right) \Bigg]$$

We use extreme learning machines to train the NN with the loss function and solve the differential equation. The Moore-Penrose inverse technique is applied to a linear set of equations using MATLAB's Pinv tool. This technique is only useful for $1 < \eta < 2$. This approach does not work with integer orders.

3.2.2 *Using Volterra approximation*

Generating the NN's LF, our approximation relies on the Volterra integral.

The corresponding form of the VIE-based approximation derived in Eq. (6) is shown below for the DE mentioned in Eq. (7)

$$\hbar(x_w) = v(x_w) + \frac{x^{-\tau(\gamma+\eta)}}{\Gamma\eta} \sum_{j=1}^{w-1} \left\{ \begin{array}{l} \left(\dfrac{F(x_j,\ \hbar(x_j))}{x_j^\tau - x_{j+1}^\tau} \right) (x_j)^{\tau\gamma} \left[\begin{array}{c} \dfrac{x_{j+1}^\tau - x_w^\tau}{\eta} \left(\begin{array}{c} (x_w^\tau - x_{j+1}^\tau)^\eta \\ -(x_w^\tau - x_j^\tau)^\eta \end{array} \right) - \dfrac{1}{\eta(\eta+1)} \\ \times \left((x_w^\tau - x_{j+1}^\tau)^{\eta+1} - (x_w^\tau - x_j^\tau)^{\eta+1} \right) \end{array} \right] \\ + \left(\dfrac{F(x_{j+1},\ \hbar(x_{j+1}))}{x_{j+1}^\tau - x_j^\tau} \right) (x_{j+1})^{\tau\gamma} \left[\begin{array}{c} \dfrac{x_j^\tau - x_w^\tau}{\eta} \left(\begin{array}{c} (x_w^\tau - x_{j+1}^\tau)^\eta \\ -(x_w^\tau - x_j^\tau)^\eta \end{array} \right) - \dfrac{1}{\eta(\eta+1)} \\ \times \left((x_w^\tau - x_{j+1}^\tau)^{\eta+1} - (x_w^\tau - x_j^\tau)^{\eta+1} \right) \end{array} \right] \end{array} \right\}.$$

Similar to the prior procedure, we now create the LF at the position x_w in the following manner:

$$L_w = \left[\begin{array}{l} \hbar(x_w) - v(x_w) - \dfrac{x^{-\tau(\gamma+\eta)}}{\Gamma\eta} \sum\limits_{j=1}^{w-1} \left\{ \begin{array}{l} \left[\left(\begin{array}{c} F(x_j,\ \hbar_{CE}(x_j),\ \hbar_{CE}(x_j - \xi_1), \\ \dfrac{\hbar_{CE}(x_j - \xi_2),\ \cdots,\ \hbar_{CE}(x_j - \xi_n))}{x_j^\tau - x_{j+1}^\tau} \end{array} \right) \right]^2 \\ (x_j)^{\tau\gamma} \left[\begin{array}{c} \left(\dfrac{x_{j+1}^\tau - x_w^\tau}{\eta} \right) \left(\begin{array}{c} (x_w^\tau - x_{j+1}^\tau)^\eta \\ -(x_w^\tau - x_j^\tau)^\eta \end{array} \right) - \dfrac{1}{\eta(\eta+1)} \\ \times \left((x_w^\tau - x_{j+1}^\tau)^{\eta+1} - (x_w^\tau - x_j^\tau)^{\eta+1} \right) \end{array} \right] \\ + \left(\begin{array}{c} F(x_j,\ \hbar_{CE}(x_j),\ \hbar_{CE}(x_j - \xi_1), \\ \dfrac{\hbar_{CE}(x_j - \xi_2),\ \cdots,\ \hbar_{CE}(x_j - \xi_n))}{x_{j+1}^\tau - x_j^\tau} \end{array} \right) \\ (x_{j+1})^{\tau\gamma} \left[\begin{array}{c} \left(\dfrac{x_j^\tau - x_w^\tau}{\eta} \right) \left(\begin{array}{c} (x_w^\tau - x_{j+1}^\tau)^\eta \\ -(x_w^\tau - x_j^\tau)^\eta \end{array} \right) - \dfrac{1}{\eta(\eta+1)} \\ \times \left((x_w^\tau - x_{j+1}^\tau)^{\eta+1} - (x_w^\tau - x_j^\tau)^{\eta+1} \right) \end{array} \right] \end{array} \right\} \end{array} \right].$$

The ideal weights and biases for the NNs are determined by minimizing the LF with respect to these terms. Now, determining the loss function by adding the interval over the whole discretized domain. As a result, the required function can be obtained by

$$\min_{\{W,\ B,\ \gamma\}} L$$

where $L = \sum_{i=1}^{t} L_i$. The solution to the FDE is approximated throughout the domain using the trained NNs with weights W, γ, and B and using an ELM-based technique, the NNs are trained.

4. Numerical results

In this section, we apply the proposed L1 and Volterra schemes to solve fractional-order equations and check their accuracy and reliability. MATLAB coding is used to verify the examples, and $\hbar_{CE}$ is used to represent the CE.

Example 1 Let us consider the differential equation

$$({}_cD_{\rho}^{\gamma,\ \alpha}Y)(x) = \frac{\Gamma(3)x^{2-\alpha}}{\Gamma(3-\alpha)} - \frac{2\Gamma(2)x^{1-\alpha}}{\Gamma(2-\alpha)} + (x^2-2x)^3 - Y^3(x),\ \ 1 < \alpha \leq 2,\ \rho \in \mathbb{R}^+,$$

subject to $Y(0) = 0$. Firstly, the domain is divided into 0.1 step size. Then, the constrained function can be written as: $Y_{CE}(x) = \mathscr{G}(x) + \rho\nu(x)$, where $\mathscr{G}(x)$ is any arbitrary function that is chosen for the NN's output. The basis function ρ has been selected so that the impact of its starting point is included in the CE. We now build the LF using the L1-based technique in the following manner:

$$L_i = \sum_{i=1}^{n} \left[\frac{1}{\rho\Gamma(3-\alpha)} \sum_{j=1}^{w-1} \left\{ \left(\frac{Y_{CE}(x_j) - Y_{CE}(x_{j+1})}{dx} \right) \left[\begin{array}{c} \left(\frac{(x_j)^{1-\rho}}{x_{j+1}^{\rho} - x_j^{\rho}} \right) \left(\begin{array}{c} (2-\alpha)(x_w^{\rho} - x_{j+1}^{\rho}) \times \\ \left((x_w^{\rho} - x_{j+1}^{\rho})^{1-\alpha} - (x_w^{\rho} - x_j^{\rho})^{1-\alpha} \right) \\ -\left((x_w^{\rho} - x_{j+1}^{\rho})^{2-\alpha} - (x_w^{\rho} - x_j^{\rho})^{2-\alpha} \right) \end{array} \right) \\ + \frac{(x_{j+1})^{1-\rho}}{x_j^{\rho} - x_{j+1}^{\rho}} \left(\begin{array}{c} (2-\alpha)(x_w^{\rho} - x_j^{\rho}) \times \\ \left((x_w^{\rho} - x_{j+1}^{\rho})^{1-\alpha} - (x_w^{\rho} - x_j^{\rho})^{1-\alpha} \right) \\ -\left((x_w^{\rho} - x_{j+1}^{\rho})^{2-\alpha} - (x_w^{\rho} - x_j^{\rho})^{2-\alpha} \right) \end{array} \right) \end{array} \right] \right\} \right]^2 .$$

$$- \left(\frac{\Gamma(3)x^{2-\alpha}}{\Gamma(3-\alpha)} - \frac{2\Gamma(2)x^{1-\alpha}}{\Gamma(2-\alpha)} + (x^2-2x)^3 - Y_{CE}^3(x) \right)$$

For $\alpha = 2$, We utilize the VIE to generate the following loss function:

$$L_w = \left[\hbar(x_w) - v(x_w) - \frac{x^{-\rho(\gamma+\alpha)}}{\Gamma\alpha} \sum_{j=1}^{w-1} \left\{ \begin{array}{l} \left(\begin{array}{c} F(x_j,\ \hbar_{CE}(x_j),\ \hbar_{CE}(x_j-\xi_1), \\ \\ \dfrac{\hbar_{CE}(x_j-\xi_2),\ \cdots,\ \hbar_{CE}(x_j-\xi_n))}{x_j^\rho - x_{j+1}^\rho} \end{array} \right) \\ (x_j)^{\rho\gamma} \left[\begin{array}{c} \left(\dfrac{x_{j+1}^\rho - x_w^\rho}{\alpha} \right) \left((x_w^\rho - x_{j+1}^\rho)^\alpha \right. \\ \left. -(x_w^\rho - x_j^\rho)^\alpha \right) - \dfrac{1}{\alpha(\alpha+1)} \\ \times \left((x_w^\rho - x_{j+1}^\rho)^{\alpha+1} - (x_w^\rho - x_j^\rho)^{\alpha+1} \right) \end{array} \right] \\ + \left(\begin{array}{c} F(x_j,\ \hbar_{CE}(x_j),\ \hbar_{CE}(x_j-\xi_1), \\ \\ \dfrac{\hbar_{CE}(x_j-\xi_2),\ \cdots,\ \hbar_{CE}(x_j-\xi_n))}{x_{j+1}^\rho - x_j^\rho} \end{array} \right) \\ (x_{j+1})^{\rho\gamma} \left[\begin{array}{c} \left(\dfrac{x_j^\rho - x_w^\rho}{\alpha} \right) \left((x_w^\rho - x_{j+1}^\rho)^\alpha \right. \\ \left. -(x_w^\rho - x_j^\rho)^\alpha \right) - \dfrac{1}{\alpha(\alpha+1)} \\ \times \left((x_w^\rho - x_{j+1}^\rho)^{\alpha+1} - (x_w^\rho - x_j^\rho)^{\alpha+1} \right) \end{array} \right] \end{array} \right\} \right]^2.$$

The L1-based LF is unable to provide the solution when $\alpha = 2$, therefore, the VIE-based LF is generated. To achieve an approximation result to the differential equation, the functions are reduced in terms of the weights applied to the hidden and output layers. In this example, we consider the fractional differential equation at fractional order $\alpha = 1.8$. To validate the accuracy of the proposed method, we perform a detailed comparison with the classical L1 numerical scheme.

The comparison is carried out in terms of both numerical values and performance metrics. Table 1 presents the exact and approximated values obtained by the proposed method and the L1 scheme. It is observed that the approximated values obtained by the proposed method are closer to the exact solution than those obtained by the L1 scheme, resulting in a smaller error. Furthermore, the accuracy is quantitatively verified using the error performance metrics RMSE, MAE, and R^2, as shown in Table 2. The results shows that the proposed method consistently achieves lower RMSE and MAE values, while the coefficient of determination R^2 is closer to 1 compared to the L1 method. This clearly demonstrates that the proposed method outperforms the L1 scheme in terms of accuracy and reliability.

The proposed method is shown to be convergent based on observed error behavior. The suggested method is convergent since the performance metrics (RMSE, MAE) gradually decrease with refinement, and since

$$\lim_{dx \to 0} E(x) = 0,$$

is satisfied by the error $E(x)$ between the precise and approximation solutions. Consequently, the outcomes of this example confirm the precision and convergence of the proposed neural network-based approach, providing a more reliable solution compared to the traditional L1 numerical method. Comparison of the exact and approximate graphs is shown in Figures 1. Figures 1a and 1b display the exact vs approximate graph using the standard L1 numerical method and

the proposed method, respectively. These figures clearly show that the approximate solution from the proposed method almost completely overlaps with the actual solution, whereas the L1 technique deviates significantly from the exact curve. For the proposed and L1 methods, the corresponding error graphs are shown in Figures 1c and 1d. The error of the L1 method clearly rises over the domain, reaching a maximum of almost 1.6×10^{-1}. On the other hand, the error of the proposed method is consistently distributed over the domain and remains extremely modest, at about 10^{-3}. These graphical comparisons thus demonstrate that the proposed method performs better than the traditional L1 numerical methodology through the provision of a significantly smaller error and a much closer approximation to the exact solution. Figure 2a presents the approximate solutions for $\alpha = 1.2$ at different ρ values. Table 3 shows the performance metrics at different ρ values which shows that the best results are obtained when $\rho = 1.1$, which also has the lowest RMSE (0.014978) and MAE (0.011351), as well as the greatest R^2 value (0.997593). The fact that the curves nearly overlap suggests that the proposed method is stable in relation to changes in ρ. The fact that the curves nearly overlap suggests that the proposed method is stable in relation to changes in ρ. The comparison between the exact and approximate solutions at $\alpha = 1.03,\ \rho = 0.5$ is shown in Figure 2b, where the results of the proposed method are almost the same as those of the exact solution. High accuracy is confirmed by the related error plot, which is displayed in Figure 2c. The greatest error stays within the range of 10^{-3}. In particular, RMSE $= 7.2184 \times 10^{-3}$, MAE $= 5.0439 \times 10^{-3}$, and $R^2 = 0.999080$ are the performance measures for this instance, which confirm the high consistency with the exact solution. Figure 2d in the same way shows the error graph for the Volterra approximation of the proposed method for $\alpha = 2,\ \rho = 0.1$. The error magnitude is incredibly small and consistent throughout the domain. The exact and approximate solutions for this example are further shown in Figure 2e, which shows almost perfect curve overlap. The performance metrics RMSE $= 2.1574 \times 10^{-3}$, MAE $= 1.4578 \times 10^{-3}$, and $R^2 = 1.0000$ demonstrate the accuracy of the proposed method.

Figure 2d in the same way shows the error graph for the Volterra approximation of the proposed method for $\alpha = 2,\ \rho = 0.1$. The error magnitude is incredibly small and consistent throughout the domain. The exact and approximate solutions for this example are further shown in Figure 2e, which shows almost perfect curve overlap. The performance metrics RMSE $= 2.1574 \times 10^{-3}$, MAE $= 1.4578 \times 10^{-3}$, and $R^2 = 1.0000$ demonstrate the accuracy of the proposed method.

Table 1. Comparison Table for Exact, Numerical (L1) [41], and proposed method for Example 1

x	Exact	Numerical (L1)	Proposed method
0	0	0	0
0.1	0.0790095924	0.0615527280	0.0788996602
0.2	0.1706247156	0.1339836787	0.1703945331
0.3	0.2785123953	0.2197133381	0.2781543435
0.4	0.4039840269	0.3195191633	0.4034921899
0.5	0.5446735850	0.4319209805	0.5440320279
0.7	0.6936846365	0.5528960648	0.6928872942
0.8	0.8401879120	0.6762439555	0.8392290659
0.9	0.9721659913	0.7947748989	0.9710394993
1.0	1.0804477603	0.9020929762	1.0791419925

Table 2. Comparison of performance metrics for Numerical (L1) and proposed method of Example 1

Method	RMSE	MAE	R^2
Numerical (L1)	1.217062e-01	1.034497e-01	0.904111
Proposed method	8.319456e-04	6.832592e-04	0.999996

Table 3. Performance metrics of the proposed method for Example 1 at $\alpha = 1.2$ for different ρ values

ρ	RMSE	MAE	R^2
0.3	0.019078	0.013193	0.996095
0.9	0.020738	0.014499	0.995386
1.1	0.014978	0.011351	0.997593
1.5	0.022592	0.015983	0.994524

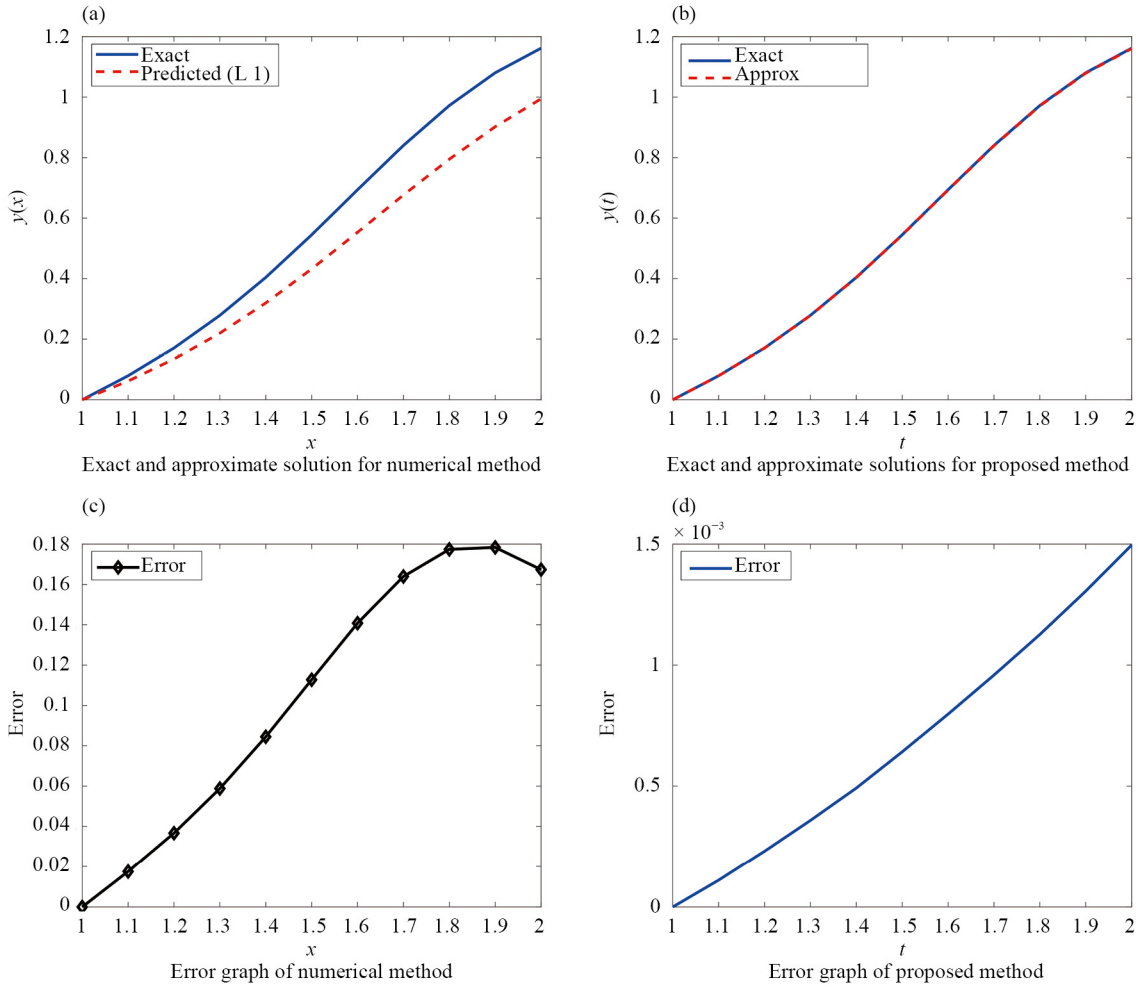


Figure 1. Plots for comparison of Example 1 using the proposed and numerical method

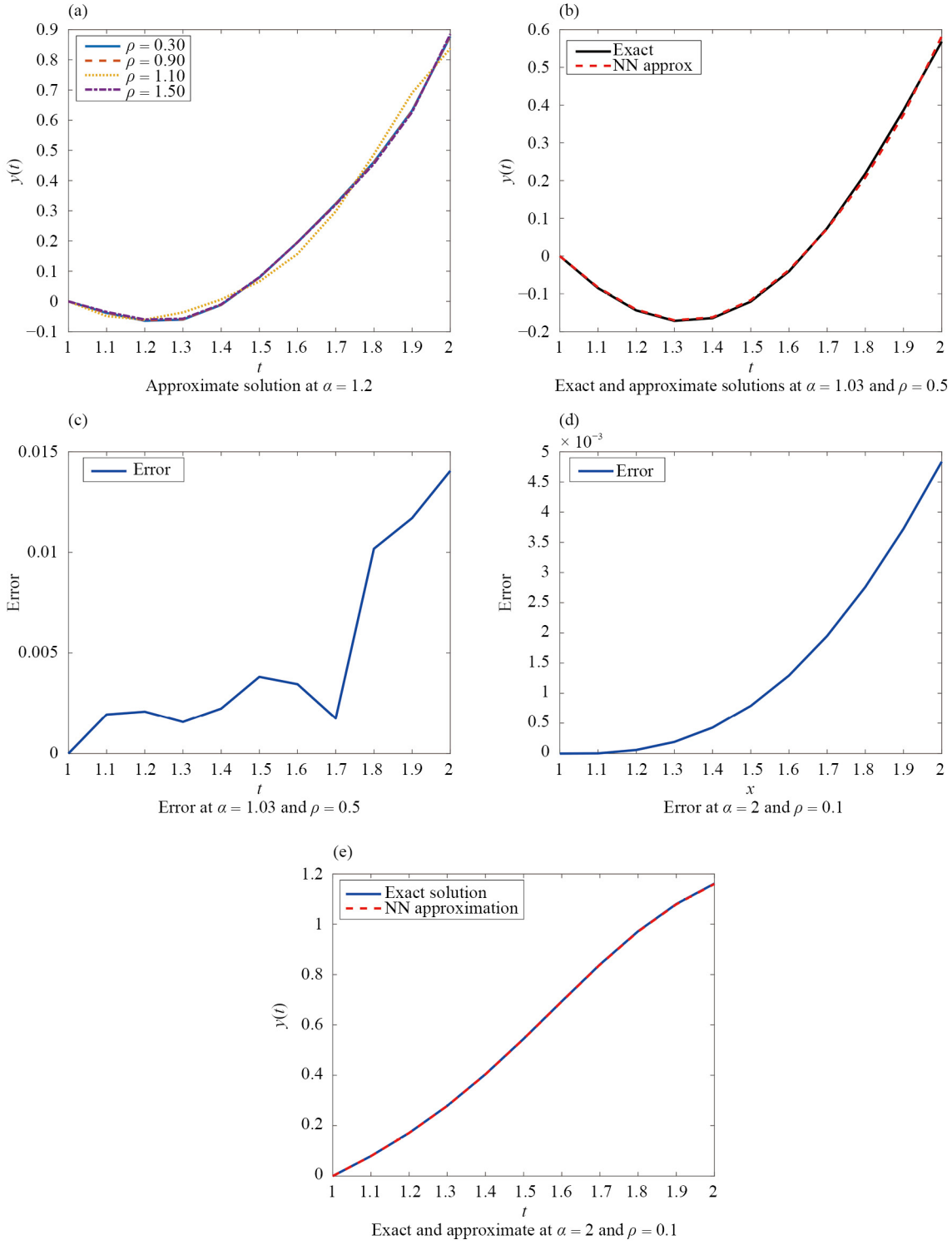


Figure 2. Plots for Example 1 using the proposed method

Example 2 In this example, the pantograph problem is solved using an Erdélyi-Kober fractional derivative. The formula is analyzed in the following way:

$$({}_cD_{\tau}^{\gamma,\,\eta}y)(x) = \frac{1}{10}y(\frac{x}{5}) - \frac{1}{10}e^{-(\frac{1}{5})x}, \quad 1 < \alpha \leq 2,\ \rho > 0,$$

subject to $y(0) = 1$. The span has been divided into 0.1-step intervals. The process is similar to that of the preceding example to generate the LF.

The approximate solutions for the fractional order $\alpha = 1.2$ are shown in Figure 3a for different values of the parameter $\rho = (0.3,\ 0.9,\ 1.5,\ 1.9)$. The results were produced using a neural network with a single hidden layer of 20 neurons, a sigmoid activation function, and optimization using the nonlinear least-squares solver lsqnonlin. Several curves that correspond to different values of ρ are closely clustered, indicating that the parameter ρ has minimal impact on the solution's behavior for this particular fractional order. The general dynamics remain rather similar despite a few little differences between the curves, especially in the area around the beginning region ($x \approx 1$). This suggests the neural network approach can be used to generate stable approximations that are not affected by small changes in ρ. Thus, Figure 3a shows that the proposed neural network structure provides reliable and consistent approximations for the fractional problem at $\alpha = 1.2$, with minimal variation across different ρ. This confirms the stability of the approach and the minimal sensitivity of the solution to the parameter ρ in this range. Now, Table 4 shows the performance metrics at different ρ values which shows that the best results are obtained when $\rho = 1.9$, which also has the lowest RMSE (0.014599) and MAE (0.012077), as well as the greatest R^2 value (0.960409). This implies that the neural network approximation closely resembles the precise solution for this parameter value. The results for $\rho = 1.5$ are also highly accurate, with RMSE and MAE values similar to those of $\rho = 1.9$ and an R^2 value of 0.954418, which still shows an excellent fit. The errors are slightly bigger with $\rho = 0.9$ and $\rho = 0.3$, with RMSE values of 0.017621 and 0.020216, respectively, and correspondingly lower R^2 values (0.942322 and 0.924082). This suggests that the approximation loses accuracy as ρ decreases, yet it still remains correct. Overall, these performance metrics demonstrate that the proposed method produces accurate results across many different kinds of ρ values, with particularly favorable outcomes for higher values such as $\rho = 1.9$. The high R^2 values and error constancy across all circumstances further show the neural network's stability and robustness. For $\alpha = 1.8$ and $\rho = 1.2$, the exact and approximate solutions are shown in Figure 3b. While the red dashed line indicates the neural network approximation produced by the proposed method, the solid black line represents the exact solution. The graphic makes it evident that the two curves nearly exactly coincide across the whole domain, indicating that the neural network is quite accurate at capturing the behavior of the exact solution.

This neural network was built using the sigmoid activation function with a single hidden layer of 20 neurons. The nonlinear least-squares solver lsqnonlin was used to optimize the parameters. Both the representational power of the neural architecture and the effectiveness of the optimization approach are demonstrated by the optimal match between the exact and approximate curves. The performance metrics provide an additional measure of the approximation's accuracy. $R^2 = 0.999995$ is the coefficient of determination, the RMSE is 1.6458×10^{-4}, and the MAE is 1.3834×10^{-4} at $\alpha = 1.8$ and $\rho = 1.2$. The neural network solution and the exact solution have nearly perfect consistency, as evidenced by these small error numbers and the R^2 value. Therefore, the performance measures validate the proposed method's dependability. The neural network may produce extremely accurate approximations for the fractional differential equation at higher fractional orders, like $\alpha = 1.8$, even with a relatively simple architecture. The error distribution between the exact and approximate solutions for $\alpha = 1.5$ and $\rho = 1.2$ is displayed in Figure 3c. Within a small magnitude of order 10^{-4}, the error fluctuates in an oscillatory manner. Since the nonlinear optimization may create small local errors, these oscillations are common when neural networks approximate fractional-order dynamics. However, the error never becomes infinite, demonstrating the stability and dependability of the suggested neural network method.

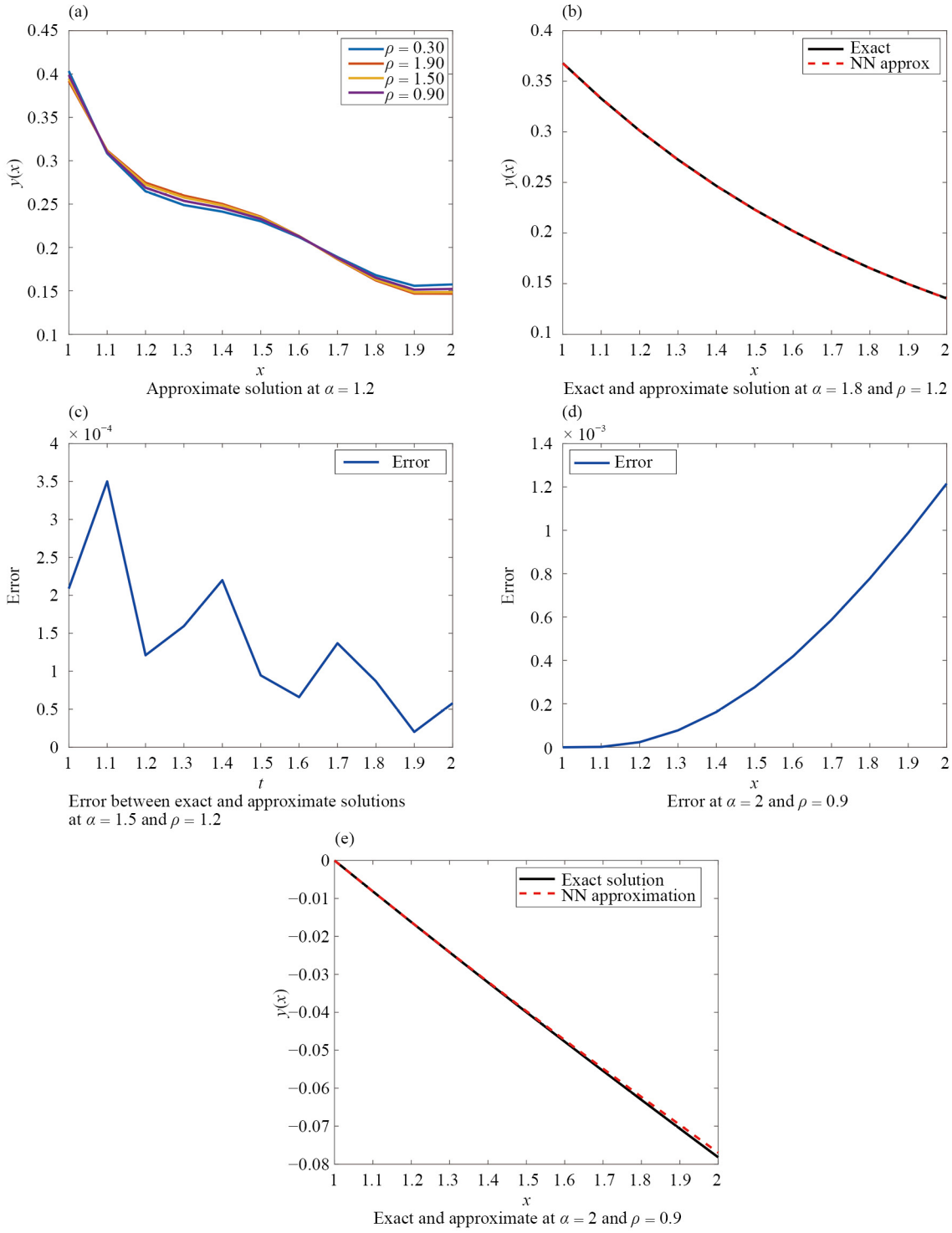


Figure 3. Plots for Example 2 using the proposed method

Table 4. Performance metrics of the proposed method for Example 2 at $\alpha = 1.2$ for different ρ values

ρ	RMSE	MAE	R^2
0.3	0.020216	0.016336	0.924082
1.9	0.014599	0.012077	0.960409
1.5	0.015665	0.012551	0.954418
0.9	0.017621	0.013728	0.942322

The error curve for the Volterra approximation of the proposed method at $\alpha = 2$ and $\rho = 0.9$ is shown in Figure 3d. In this case, the error progressively rises with x, although its size remains modest at 2.0×10^{-3}. This steady increase in error suggests that there are no unexpected instabilities and that the approximation is constant throughout the region. The exact and approximate solutions for $\alpha = 2$ and $\rho = 0.9$ are compared in Figure 3e. The dashed red curve, which represents the neural network approximation, and the solid black curve, which represents the exact solution, nearly overlap. The two curves' near-indistinguishability demonstrates that the neural network can accurately capture the dynamics of the integer-order problem even with a single hidden layer of 20 neurons with sigmoid activation. At $\alpha = 2$ and $\rho = 0.9$, the effectiveness of the proposed method is measured by the following metrics: MAE = 4.1174×10^{-4}, $R^2 = 0.9995$, and RMSE = 5.7911×10^{-4}. These values demonstrate good precision because both error metrics are quite small and the coefficient of determination is very close to unity. Thus, the suggested method, a neural network approximation of the Volterra type, proves the robustness even in the limiting integer-order scenario.

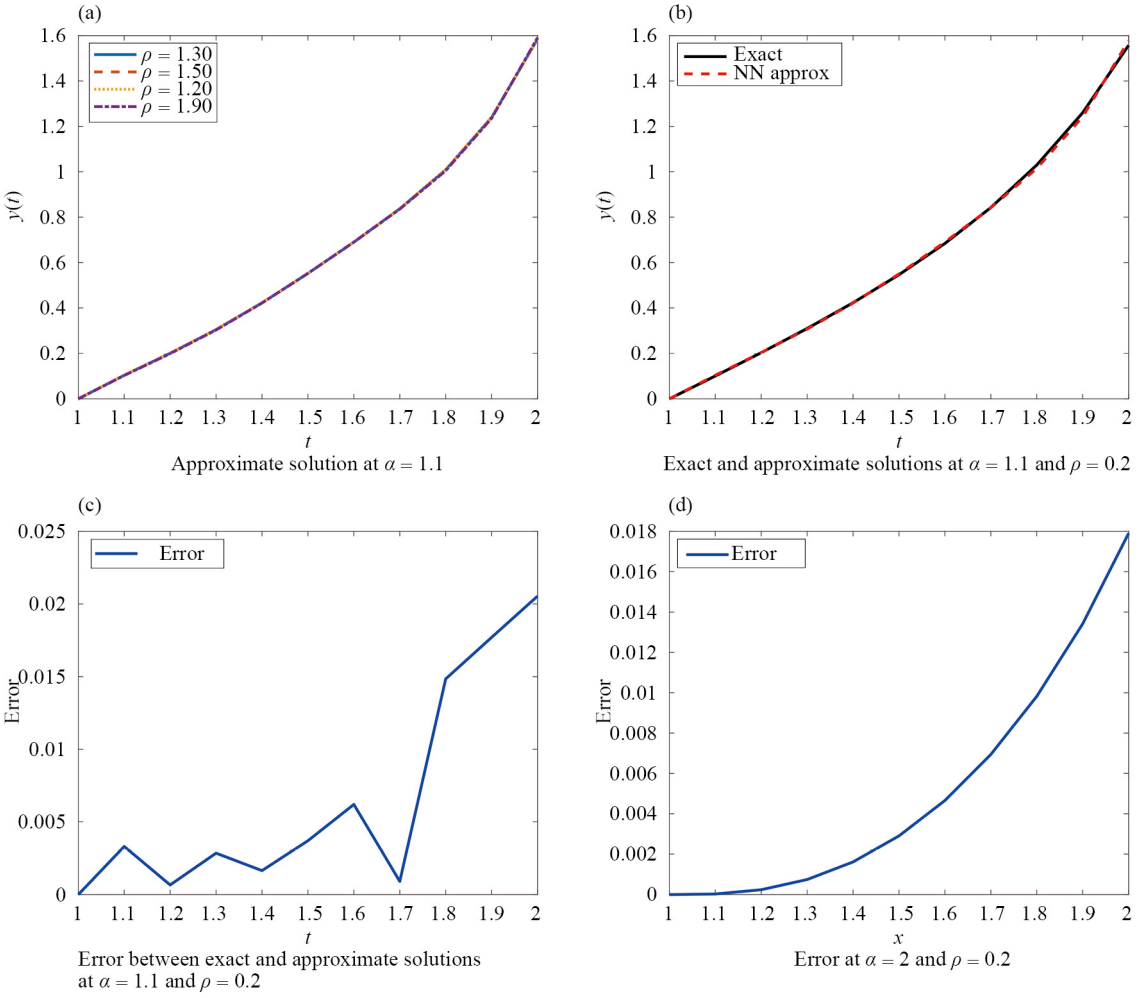
Example 3 Let us consider the generalised Caputo FDE

$$({}_cD_{\rho}^{\gamma,\,\alpha}Y)(x) = y^2 + 1, \;\; 1 < \alpha \leq 2, \;\; \rho \in \mathbb{R}^+$$

subject to $y(0) = 0$.

The ideal solution is obtained by minimizing the LF with respect to the weights of the NN. For using the NN based scheme, here we take a step size 0.1. The approximate solutions of the nonlinear fractional differential equation at different values of the parameter ρ are displayed in Figure 4a for the function $f(y) = y^2 + 1$ for $\alpha = 1.1$. The neural network used in this case is the proposed L1 neural network-based method, which features a single hidden layer of 20 neurons with sigmoid activation and is optimized with lsqnonlin. For $\rho = 1.2,\ 1.3,\ 1.5$, and 1.9, the figure displays approximate solutions. The neural network approximation is stable and consistent throughout a range of values of ρ, as evidenced by the nearly perfect overlap of all curves. This overlap highlights the robustness of the NN-based technique and shows that the proposed approach yields reliable results even when the parameter ρ varies. The nonlinear source term $y^2 + 1$ is reflected in the smooth monotonic growth of $y(t)$, and the close agreement between curves suggests that ρ has a minimal impact on the solution behavior for this fractional order. Three popular metrics R^2, MAE, and RMSE are used in Table 5 to characterize the accuracy of the proposed NN-based technique at $\alpha = 1.1$ for different ρ values. With RMSE = 0.011982, MAE= 0.008380, and $R^2 = 0.999362$, the method works best at $\rho = 1.2$. With R^2 consistently above 0.998 and errors remaining extremely small (order of 10^{-2}), the performance remains good for further values of ρ $(1.3,\ 1.5,\ 1.9)$, indicating a very strong relationship between exact and approximate solutions. A slight increase in RMSE and MAE at $\rho = 1.9$ $(0.015088,\ 0.010499)$ suggests that approximation accuracy slightly decreases at higher ρ levels, even though the results are still within an appropriate range. Over the interval $\alpha \in (1,\ 2]$, the Figure 4b compares the exact and approximate solutions of a differential equation. The red dashed line indicates the estimated solution derived from a neural network, whereas the solid black curve indicates the precise solution. The proposed method, which is based on a neural network with a single hidden layer that has 20 neurons and a sigmoid activation function, is specifically used to compute the approximation. The proposed method's ability to accurately capture the behavior of the underlying system is demonstrated by the close agreement between the exact and NN-based solutions. The figure represents a fractional

or nonlinear framework with $\alpha = 1.1$ and $\rho = 0.2$. For $\alpha = 1.1$ and $\rho = 0.2$, the error graph 4c shows generally good accuracy, with a peak near the right border and some fluctuations. The difference between the precise and approximate solutions is still very small, mostly ranging between 10^{-3} and 10^{-2}. The R^2 value of 0.9996, along with the performance metrics of RMSE and MAE of 9.68×10^{-3} and 6.58×10^{-3}, respectively, shows that the neural network approximation and the exact solution fit each other reasonably well. The error plot 4d for $\alpha = 2$ and $\rho = 0.2$ using the Volterra-based neural network shows a smooth, monotonic growth from nearly zero to nearly 1.6×10^{-2}. However, the comparison plot of the exact and approximate solutions 4e shows that the two curves almost overlap, indicating high accuracy. The performance metrics further support this, indicating that the Volterra-based method provides a better approximation for $\alpha = 2$, with an improved R^2 of 0.9997 and RMSE and MAE dropping to 7.85×10^{-3} and 5.30×10^{-3}, respectively.



Approximate solution at $\alpha = 1.1$

Exact and approximate solutions at $\alpha = 1.1$ and $\rho = 0.2$

Error between exact and approximate solutions at $\alpha = 1.1$ and $\rho = 0.2$

Error at $\alpha = 2$ and $\rho = 0.2$

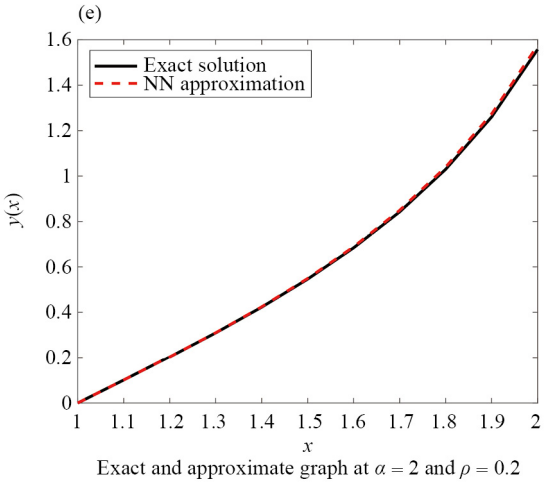


Figure 4. Plots for Example 3 using the proposed method

Table 5. Performance metrics of the proposed method for Example 3 at $\alpha = 1.1$ for different ρ values

ρ	RMSE	MAE	R^2
1.3	0.012339	0.008624	0.999323
1.5	0.013109	0.009150	0.999236
1.2	0.011982	0.008380	0.999362
1.9	0.015088	0.010499	0.998989

Example 4 This example shows the delay differential equation using Erdélyi-Kober fractional derivative as follows:

$$({}_cD_{\tau}^{\gamma,\,\eta}y)(x) = \frac{by(x-\xi)}{1+[y(x-\xi)]^n} - ay(x),$$

where $x < 0$, $1 < \eta \leq 2$, $\tau > 0$. Taking parameters as $a = 0.1,\ b = 0.2,\ n = 10,\ \xi = 10$, with a step size of 0.1. Now, we construct the loss function with a multiple delay using L1 based approximation as follows:

$$L_i = \sum_{i=1}^{n} \left[\frac{1}{\tau\Gamma(3-\eta)} \sum_{j=1}^{w-1} \left\{ \left(\frac{\hbar_{CE}(x_j) - \hbar_{CE}(x_{j+1})}{dx} \right) \left[\begin{array}{c} \left(\frac{(x_j)^{1-\tau}}{x_{j+1}^{\tau} - x_j^{\tau}} \right) \left(\begin{array}{c} (2-\eta)(x_w^{\tau} - x_{j+1}^{\tau}) \times \\ \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} - (x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ -\left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} - (x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{array} \right) \\ + \frac{(x_{j+1})^{1-\tau}}{x_j^{\tau} - x_{j+1}^{\tau}} \left(\begin{array}{c} (2-\eta)(x_w^{\tau} - x_j^{\tau}) \times \\ \left((x_w^{\tau} - x_{j+1}^{\tau})^{1-\eta} - (x_w^{\tau} - x_j^{\tau})^{1-\eta} \right) \\ -\left((x_w^{\tau} - x_{j+1}^{\tau})^{2-\eta} - (x_w^{\tau} - x_j^{\tau})^{2-\eta} \right) \end{array} \right) \end{array} \right] \right\} \right]^2 .$$

$$- \left(\frac{by(x-\xi)}{1+[y(x-\xi)]^n} - ay(x) \right)$$

Figure 5a presents a comparison between the exact and approximate solutions of the multiple-delay fractional differential equation. The approximate solution derived using the proposed L1-based neural network approximation is shown by the red dashed line, while the actual solution is represented by the black solid line. The exceptional accuracy and dependability of the neural network approximation in capturing the actual dynamics of the system are demonstrated by the two curves' nearly perfect overlap across the entire domain. As x grows, the solution shows a monotonic decay trend that is consistent with the expected properties of nonlinear delay-driven systems, beginning with an initial value close to $y(0) \approx 0.1$ and progressively approaching zero. The strong consistency between the approximation and exact results shows that the L1-based neural network approach manages the problem's delay and nonlinearity well, offering a reliable solution for fractional differential equations. The Figure 5b shows the error analysis between the exact and approximate solutions of the multiple-delay fractional differential equation solved using the L1-based neural network approximation.

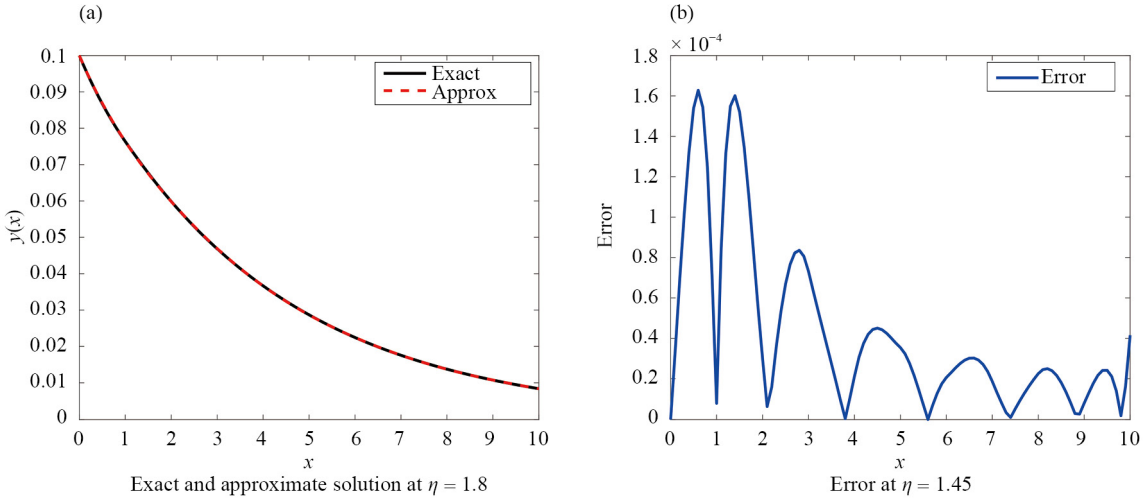


Figure 5. Plots for Example 4 using the proposed method

The plot shows that the error is still very small, at about 10^{-4}, indicating that the approximation method is very accurate. The error first exhibits oscillatory peaks, reaching its highest at $x \approx 1$ at about 1.6×10^{-4}. The oscillations' gradual amplitude drop as x increases and the error stabilizing at even lower levels demonstrate how successfully the neural network approximation converges to the exact solution over the domain. Overall, the figure shows that the approximation error is small, illustrating how well the L1-based neural network handles fractional dynamics nonlinear delay systems. The method achieved very low error values i.e RMSE = 5.939105×10^{-5}, MAE = 4.214400×10^{-5} and a coefficient of determination close to unity $R^2 = 0.999994$ confirming the reliability of the approximation.

5. Conclusions

In this study, we addressed the numerical solution of differential equations involving the EK fractional derivative. A VIE formulation was derived to train the NNs, alongside the design of an appropriate LF. The LF was minimized using the TFC and PINNs, ensuring the accurate satisfaction of the governing equations. To further enhance training efficiency and mitigate overfitting, the ELM approach was employed, significantly improving computational speed. The error analysis demonstrates the accuracy and reliability of the proposed framework. Additionally, a minimization strategy was adopted to optimize the LF, producing high-fidelity solutions for the EK fractional derivative model. The results of a comparison between the proposed L1-based NN method and the traditional L1 numerical scheme is shown in example 1 demonstrated the extremely high accuracy and resilience of the proposed method. Then the framework was successfully extended to handle multiple-delay problems involving the EK fractional derivative, illustrating the

robustness and broad applicability of the proposed methods. In the last, for all the illustrative examples in this study, we have presented the performance metrics (RMSE, MAE, R^2), proving the robustness, consistency, and dependability of the proposed approach for fractional-order systems. This study has demonstrated the proposed methodology using the framework of the EK derivative. Nevertheless, it should be mentioned that the method is not limited to this operator alone. Because the neural network-based approximation and the construction of constrained expressions are independent of the specific kernel of the derivative, the method can be readily extended to other fractional derivatives, such as the Caputo, Caputo-Fabrizio, Atangana-Baleanu, and Riemann-Liouville derivatives, with the right modifications. This shows that the proposed method is flexible and may be applied to a larger class of fractional operators in the literature. There are several directions for future research. The extension of the proposed technique to additional fractional derivatives, such as Caputo, Caputo-Fabrizio, Atangana-Baleanu, and Riemann-Liouville, would be more useful and show the generality of the methodology. Training efficiency and stability could be improved by utilizing adaptive optimization strategies and complex regularization techniques. Using the paradigm for higher-dimensional, connected, or nonlinear fractional systems could provide deeper insights into practical applications. A key step in strengthening the mathematical foundations of the proposed method will be the creation of a comprehensive convergence and error analysis.

Confilict of interest

The authors declare no competing financial interest.

References

[1] Pakdaman M, Ahmadian A, Effati S, Salahshour S, Baleanu D. Solving differential equations of fractional order using an optimization technique based on training artificial neural network. *Applied Mathematics and Computation*. 2017; 293: 81-95. Available from: https://doi.org/10.1016/j.amc.2016.07.021.

[2] Agarwal S, Mishra LN. Attributes of residual neural networks for modeling fractional differential equations. *Heliyon*. 2024; 10(19): e38332. Available from: https://doi.org/10.1016/j.heliyon.2024.e38332.

[3] Mai T, Mortari D. Theory of functional connections applied to quadratic and nonlinear programming under equality constraints. *Journal of Computational and Applied Mathematics*. 2022; 406: 113912. Available from: https://doi.org/10.1016/j.cam.2021.113912.

[4] Schiassi E, Furfaro R, Leake C, Florio MD, Johnston H, Mortari D. Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations. *Neurocomputing*. 2021; 457: 334-356. Available from: https://doi.org/10.1016/j.neucom.2021.06.015.

[5] Pathak VK, Mishra LN, Mishra VN. On the solvability of a class of nonlinear functional integral equations involving Erdélyi-Kober fractional operator. *Mathematical Methods in the Applied Sciences*. 2023; 46(13): 14340-14352. Available from: https://doi.org/10.1002/mma.9322.

[6] Farid G, Mishra VN. Caputo fractional derivative inequalities via $(h-m)$-convexity. *Journal of Mathematical Extension*. 2020; 16(5): 1-15. Available from: https://doi.org/10.30495/JME.2022.1349.

[7] Hu S, Khavanin M, Zhuang W. Integral equations arising in the kinetic theory of gases. *Applicable Analysis*. 1989; 34(3-4): 261-266. Available from: https://doi.org/10.1080/00036818908839899.

[8] Darwish MA. On Erdélyi-Kober fractional Urysohn-Volterra quadratic integral equations. *Applied Mathematics and Computation*. 2016; 273: 562-569. Available from: https://doi.org/10.1016/j.amc.2015.10.040.

[9] Agila A, Baleanu D, Eid R, Irfanoglu B. A freely damped oscillating fractional dynamic system modeled by fractional Euler-Lagrange equations. *Journal of Vibration and Control*. 2018; 24(7): 1228-1238. Available from: https://doi.org/10.1177/1077546316685228.

[10] Sun HG, Zhang Y, Baleanu D, Chen W, Chen Y. A new collection of real world applications of fractional calculus in science and engineering. *Communications in Nonlinear Science and Numerical Simulation*. 2018; 64: 213-231. Available from: https://doi.org/10.1016/j.cnsns.2018.04.019.

[11] Ansari M, Mishra LN. Common solution to a coupled system of fractional differential equations and nonlinear integral equations via weakly altering distance functions and w-distance. *Advances in Studies: Euro-Tbilisi Mathematical Journal*. 2025; 18(2): 149-174. Available from: https://doi.org/10.1186/s13662-019-2035-2.

[12] Ghafoor A, Fiaz M, Hussain M, Ullah A, Ismail EA, Awwad FA. Dynamics of the time-fractional reaction-diffusion coupled equations in biological and chemical processes. *Scientific Reports*. 2024; 14: 7549. Available from: https://doi.org/10.1038/s41598-024-58073-z.

[13] Suzuki JL, Gulian M, Zayernouri M, D'Elia M. Fractional modeling in action: A survey of nonlocal models for subsurface transport, turbulent flows, and anomalous materials. *Journal of Peridynamics and Nonlocal Modeling*. 2023; 5(3): 392-459. Available from: https://doi.org/10.1007/s42102-022-00085-2.

[14] Paul SK, Mishra LN. Asymptotic stability and approximate solutions to quadratic functional integral equations containing ψ-Riemann-Liouville fractional integral operator. *Computational Mathematics and Mathematical Physics*. 2025; 65(2): 320-338. Available from: https://doi.org/10.1134/S096554252470204X.

[15] Farid G, Bibi S, Rathour L, Mishra LN, Mishra VN. Fractional versions of Hadamard inequalities for strongly (s, m)-convex functions via Caputo fractional derivatives. *Korean Journal of Mathematics*. 2023; 31(1): 75-94. Available from: https://doi.org/10.11568/kjm.2023.31.1.75.

[16] Odibat Z, Baleanu D. Numerical simulation of initial value problems with generalized Caputo-type fractional derivatives. *Applied Numerical Mathematics*. 2020; 156: 94-105. Available from: https://doi.org/10.1016/j.apnum.2020.04.015.

[17] Okposo NI, Raghavendar K, Khan N, Agullar JFG, Jonathan AM. New exact optical solutions for the Lakshmanan-Porsezian-Daniel equation with parabolic law nonlinearity using the ϕ^6-expansion technique. *Nonlinear Dynamics*. 2025; 113(5): 4775-4795. Available from: https://doi.org/10.1007/s11071-024-10430-3.

[18] Raghavendar K, Pavani K, Aruna K, Okposo NI, Inc M. Application of natural transform decomposition method for solution of fractional Richards equation. *Contemporary Mathematics*. 2024; 5(4): 5881-5900. Available from: https://doi.org/10.37256/cm.5420245314.

[19] Yadav AK, Pandey RM, Mishra VN, Agarwal R. Some integral inequalities involving a fractional integral operator with extended hypergeometric function. *Dolomites Research Notes on Approximation*. 2025; 18(1): 17-26. Available from: https://doi.org/10.25430/pupj-DRNA-2025-1-2.

[20] Zhang K. Applications of Erdélyi-Kober fractional integral for solving time-fractional Tricomi-Keldysh type equation. *Fractional Calculus and Applied Analysis*. 2020; 23(5): 1381-1400. Available from: https://doi.org/10.1515/fca-2020-0068.

[21] Jena BB, Paikray SK, Mohiuddine SA, Mishra VN. Relatively equi-statistical convergence via deferred Nörlund mean based on difference operator of fractional-order and related approximation theorems. *AIMS Mathematics*. 2020; 5(1): 650-672. Available from: https://doi.org/10.3934/math.2020044.

[22] Hyder AA, Barakat MA. Novel improved fractional operators and their scientific applications. *Advances in Difference Equations*. 2021; 389: 1-24. Available from: https://doi.org/10.1186/s13662-021-03547-x.

[23] Hanna LM, Luchko YF. Operational calculus for the Caputo-type fractional Erdélyi-Kober derivative and its applications. *Integral Transforms and Special Functions*. 2014; 25(5): 359-373. Available from: https://doi.org/10.1080/10652469.2013.856901.

[24] Arioua Y, Titraoui M. New class of boundary value problem for nonlinear fractional differential equations involving Erdélyi-Kober derivative. *Communications in Mathematics*. 2019; 27(2): 113-141. Available from: https://doi.org/10.2478/cm-2019-0011.

[25] Hussein GA, Ziane D. Solving biological population model by using FADM within Atangana-Baleanu fractional derivative. *Journal of Education for Pure Science*. 2024; 14(2): 77-88. Available from: https://doi.org/10.32792/jeps.v14i2.434.

[26] Baleanu D, Jassim HK. Exact solution of two-dimensional fractional partial differential equations. *Fractal and Fractional*. 2020; 4(2): 21. Available from: https://doi.org/10.3390/fractalfract4020021.

[27] Toufik M, Atangana A. New numerical approximation of fractional derivative with non-local and non-singular kernel: application to chaotic models. *European Physical Journal Plus*. 2017; 132: 444. Available from: https://doi.org/10.1140/epjp/i2017-11717-0.

[28] Singh K, Singh T, Mishra LN, Dubey R, Rathour L. A brief review of predator-prey models for an ecological system with a different type of behaviors. *Korean Journal of Mathematics*. 2024; 32(3): 381-406. Available from: https://doi.org/10.11568/kjm.2024.32.3.381.

[29] Monteiro NZ, Mazorche SR. Fractional derivatives applied to epidemiology. *Trends in Computational and Applied Mathematics*. 2021; 22(2): 157-177. Available from: https://doi.org/10.5540/tcam.2021.022.02.00157.

[30] Baghdadi G, Jafari S, Sprott JC, Towhidkhah F, Golpayegani MRH. A chaotic model of sustaining attention problem in attention deficit disorder. *Communications in Nonlinear Science and Numerical Simulation*. 2015; 20(1): 174-185. Available from: https://doi.org/10.1016/j.cnsns.2014.05.015.

[31] Erturk VS, Godwe E, Baleanu D, Kumar P, Asad J, Jajarmi A. Novel fractional-order Lagrangian to describe motion of beam on nanowire. *Acta Physica Polonica A*. 2021; 140: 265-272. Available from: https://doi.org/10.12693/APhysPolA.140.265.

[32] Torre A. The fractional Fourier transform and some of its applications to optics. *Progress in Optics*. 2002; 43: 531-596. Available from: https://doi.org/10.1016/S0079-6638(02)80031-2.

[33] Qureshi S, Aziz S. Fractional modeling for a chemical kinetic reaction in a batch reactor via nonlocal operator with power law kernel. *Physica A*. 2020; 542: 123494. Available from: https://doi.org/10.1016/j.physa.2019.123494.

[34] Bhat IA, Mishra LN, Mishra VN. Comparative analysis of nonlinear Urysohn functional integral equations via Nyström method. *Applied Mathematics and Computation*. 2025; 494: 129287. Available from: https://doi.org/10.1016/j.amc.2025.129287.

[35] Bouteraa N, Inc M, Hashemi MS, Benaicha S. Study on the existence and nonexistence of solutions for a class of nonlinear Erdélyi-Kober type fractional differential equation on unbounded domain. *Journal of Geometry and Physics*. 2022; 178: 104546. Available from: https://doi.org/10.1016/j.geomphys.2022.104546.

[36] Luchko Y, Trujillo J. Caputo-type modification of the Erdélyi-Kober fractional derivative. *Fractional Calculus and Applied Analysis*. 2007; 10(3): 249-267.

[37] Mortari D, Furfaro R. Univariate theory of functional connections applied to component constraints. *Mathematical and Computational Applications*. 2021; 26(1): 9. Available from: https://doi.org/10.3390/mca26010009.

[38] Guo L, Wu H, Yu X, Zhou T. Monte Carlo fPINNs: Deep learning method for forward and inverse problems involving high dimensional fractional partial differential equations. *Computer Methods in Applied Mechanics and Engineering*. 2022; 400: 115523. Available from: https://doi.org/10.1016/j.cma.2022.115523.

[39] Schiassi E, Florio MD, Ganapol BD, Picca P, Furfaro R. Physics-informed neural networks for the point kinetics equations for nuclear reactor dynamics. *Annals of Nuclear Energy*. 2022; 167: 108833. Available from: https://doi.org/10.1016/j.anucene.2021.108833.

[40] Huang GB, Zhu QY, Siew CK. Extreme learning machine: Theory and applications. *Neurocomputing*. 2006; 70(1-3): 489-501. Available from: https://doi.org/10.1016/j.neucom.2005.12.126.

[41] Kundaliya PJ, Chaudhary S. Symmetric fractional order reduction method with L1 scheme on graded mesh for time fractional nonlocal diffusion-wave equation of Kirchhoff type. *Computers and Mathematics with Applications*. 2023; 149: 128-134. Available from: https://doi.org/10.1016/j.camwa.2023.08.031.