Boundary Value Problems
a SpringerOpen Journal

RESEARCH Open Access

An application of artificial neural networks for solving fractional higher-order linear integro-differential equations

T. Allahviranloo[1], A. Jafarian[2], R. Saneifard[2], N. Ghalami[2], S. Measoomy Nia[2], F. Kiani[3], U. Fernandez-Gamiz[4] and S. Noeiaghdam[5,6*]

*Correspondence: snoei@istu.edu; noiagdams@susu.ru
[5]Industrial Mathematics Laboratory, Baikal School of BRICS, Irkutsk National Research Technical University, Irkutsk, 664074, Russia
[6]Department of Applied Mathematics and Programming, South Ural State University, Lenin prospect 76, Chelyabinsk, 454080, Russia
Full list of author information is available at the end of the article

Abstract

This ongoing work is vehemently dedicated to the investigation of a class of ordinary linear Volterra type integro-differential equations with fractional order in numerical mode. By replacing the unknown function by an appropriate multilayered feed-forward type neural structure, the fractional problem of such initial value is changed into a course of non-linear minimization equations, to some extent. Put differently, interest was sparked in structuring an optimized iterative first-order algorithm to estimate solutions for the origin fractional problem. On top of that, some computer simulation models exemplify the preciseness and well-functioning of the indicated iterative technique. The outstanding accomplished numerical outcomes conveniently reflect the productivity and competency of artificial neural network methods compared to customary approaches.

Keywords: Higher-order linear integro-differential equation; Artificial neural network approach; Caputo fractional derivative; Learning algorithm; Cost function

1 Introduction

To the best of our knowledge, the expansion of the notion of differentiation and integration to random non-integer (real/complex) order formed with the well-established foundation of deliberations on fractional calculus. Over the course of the last two decades, many researchers have studied fractional calculus in the domain of modern mathematics. That is why the problem of fractional-order integro-differential equations (FOIDEs) is generally utilized in applied mathematics in the same way as in other linked domains of science and engineering. This is one of the foremost reasons why the issue tackled in the current study has been more interesting to a large circle of researchers and scientists. It is well known that a majority of the initial or boundary value problems in fractional derivatives are not meant to be solved explicitly. A class of non-linear fractional Fredholm–Volterra–Hammerstein integro-differential delay equations with a functional bound was studied by Kurkcu [14]. Wnang [23] has developed a hybrid method based on the combination of Bernoulli polynomials approximation and Caputo fractional derivative and numerical integral transformation to approximate solutions of two-dimensional

non-linear Volterra–Fredholm integral equations and fractional integro-differential equations (of Hammerstein and mixed types). Nematia and Lima showed a case in point for an adjustment of hat functions in finding solutions to a class of non-linear singular fractional integro-differential equations [17]. Elbeleze et al. [5] applied the methods of homotopy perturbation and variational iteration for Fredholm–Volterra type integro-differential equations with initial/boundary conditions. Wang and Zhu [24] employed Euler wavelet approximation to gain a solution to non-linear fractional-order Volterra type integro-differential equations. Bazgir and Ghazanfari [2] benefited from a very efficient combination of a self-adjoint operator at fourth order concerning a fractional order of eigenfunctions and changed the Legendre polynomials into the numerical solution for a fourth-order fractional partial integro-differential equation. Rahimkhani et al. [19] used the Bernoulli pseudo-spectral method to solve non-linear fractional Volterra integro-differential equations. A large number of numerical methods deal with solutions for several types of fractional integro-differential equations. For more details, the reader is referred to references [1, 7, 25, 26]. It is noteworthy that Bentrcia et al. [3] investigated the asymptotic stability of a viscoelastic Bresse system in the one-dimensional bounded domain. They introduced two internal damping terms expressed using the generalized Caputo fractional derivative. Of course, it seems necessary to mention that Mennouni [15] established an improved convergence analysis via the Kulkarni method to approximate the solution of an integro-differential equation in $L^2([-1,1])$.

In order to model and solve newly emerging and complex mathematic problems, it is highly recommended to employ the approach of artificial neural networks (ANNs), which simulate the neural structure of the human brain, which has been called one of the world's wonders. Of course, it is highly desirable to recall that the multiple structures of these networks were previously used to estimate solutions of different mathematical problems in applied grounds (for instance, see [9, 10, 22]). Also, remarkable fractional-order mathematical problems have been numerically examined through the ANN approach in the recent past [11–13, 21]. Now, an appropriate structure of ANNs will be introduced and later applied to numerically solve a fractional higher-order linear Volterra type integro-differential equation having initial conditions. In this regard, in order to model the fractional problem in question, an appropriate three-layered feed-forward neural network will be designed and then applied. The neural architecture, based on a first-order gradient descent optimization algorithm, is able to transform the origin fractional problem into a minimization one through accumulating the initial grounds. Then a back-propagation (BP) algorithm is used to train the designed network, until the network error reaches an acceptable value. Now, the supposed ANN architecture is able to estimate the unknown function on a solution area to any desired accuracy. The present paper is organized as follows: In Sect. 2, different notations and definitions used in fractional calculus and ANNs are shortly elaborated on. In Sect. 3, an acceptable architecture of neural networks is structured for estimating solutions of the mentioned fractional problem. In Sect. 4, numerical accomplishments are appropriately illustrated to indicate the accuracy of the proposed iterative technique. Finally, in Sect. 5 the discussion is extended with the major outcomes of the recommended method.

2 Preliminaries

As declared before, the preeminent goal of the current research is to apply the ANN approach to approximate the solution of a FOIDE problem. The current section provides a

clear explanation of several required mathematical interpretations, features of fractional calculus theory, and the ANN approach.

2.1 Fractional calculus

A short summary of the literature on fractional calculus reveals that the field initiated with the question "how can a function's derivative and integral be generalized to a non-integer order?" Following this ambiguous controversy, the mathematical explanation of a non-integer-order integral or derivative was placed under the spotlight by a number of scholars for a specific period of time. Finally, Lacroix presented the first research concerning fractional-order derivatives [20]. Over the course of the following years, numerous researchers studied the subject of fractional calculus and proffered multifarious applicable descriptions of non-integer-order derivatives or integrals. Amongst the definitions, the Caputo and Riemann–Liouville definitions seem to be the two most used ones. There is no need to note which derivative has been utilized more widely, as each derivative has its own appropriate operational range. The Caputo definition more appropriately describes problems of initial value fractional order [27]. Due to the congruence of the initial conditions, we decided to use Caputo's fractional definition in this research. The Caputo fractional differential operator, proposed by the Italian mathematician Caputo [4], is clearly defined below.

Definition 1 Let $u(x)$ be a continuously differentiable function on finite interval $[a,b]$ up to order k. The Caputo derivative D_x^α and fractional integral operator $I_{a,x}^\alpha$ of order $\alpha > 0$ are defined as follows:

$$ {}_aD_x^\alpha\big[u(x)\big] = \begin{cases} \frac{d^k u(x)}{dx^k}, & \alpha = k \in N, \\ \frac{1}{\Gamma(k-\alpha)} \int_a^x \frac{u^{(k)}(\tau)}{(x-\tau)^{\alpha-k+1}}\, d\tau, & x > a, \quad 0 \le k-1 < \alpha < k, \end{cases} \tag{1} $$

$$ I_{a,x}^\alpha\big[u(x)\big] = \frac{1}{\Gamma(\alpha)} \int_a^x \frac{u(\tau)}{(x-\tau)^{1-\alpha}}\, d\tau, \tag{2} $$

respectively. Many studies have been conducted on the properties and performance of the Caputo fractional operator. Here, we will focus on its important properties and uses. It should be noted that the derivative of any order of the constant function is zero and also that the following attributes hold:

$$ {}_aD_x^\alpha\big[x^k\big] = \begin{cases} 0, & k \in Z^+, k < \lceil \alpha \rceil, \\ \frac{\Gamma(k+1)}{\Gamma(k+1-\alpha)} x^{k-\alpha}, & x > a, \quad k \in Z^+, k \ge \lceil \alpha \rceil, \end{cases} \tag{3} $$

$$ I_{0,x}^\alpha\big[t^k\big] = \frac{\Gamma(k+1)}{\Gamma(k+1+\alpha)} x^{k+\alpha}, \quad k \in Z^+. \tag{4} $$

In the above relations, the notation $\lceil \alpha \rceil$ indicates the smallest integer greater than or equal to constant α.

2.2 Basic structure of ANNs

As time passes, new methods are developed and others fall into disfavor. As is well known, an ANN can be defined as a tidy brain-inspired computing system intended to empower computer systems to learn empirically. The logic of the ANN procedure is to gather some

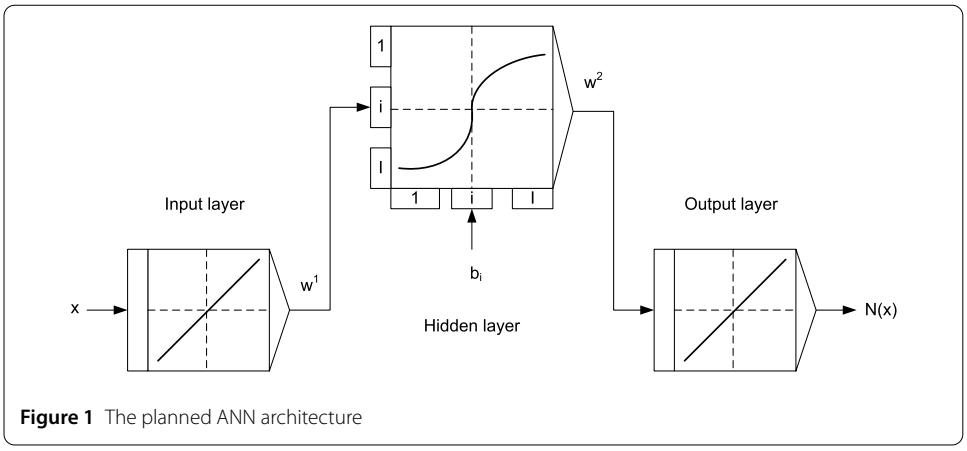

Figure 1 The planned ANN architecture

training figures and then automatically establish a system that is capable of learning from the training data. Based on such perspective, we will consider a neural architecture called "perceptron," which was suggested by the researcher Frank Rosenblatt, who founded an important research program. In such a neural network, various numbers of signals are introduced to input neurons. The network's first layer (known as the input layer) does not change the input signals' values. Neurons in the second layer (called the hidden layer) combine their inputs by benefiting from a set of network weights and biases. Then they pass through the nodes of the hidden layer via a proper activation function. Here, the sigmoidal activation function is employed to control the oscillation of the hidden neurons' output. The output of each node in the hidden layer is then passed on to the last layer of neurons to generate the output of the network. One should bear in mind that this model, just as many others, utilizes the identity function for the input and output layers. For more details on the proposed approach, see [6, 8]. Paying attention to the neural architecture illustrated in Fig. 1, one can comprehend the usefulness and innovative value of ANNs. The description elaborated on in this part can be formulated as follows:

- *input layer unit:*

$$o_1^1 = x; \tag{5}$$

- *hidden layer units:*

$$o_i^2 = f(net_i), \quad i = 1, \ldots, I, \tag{6}$$
$$net_i = x.w_i^1 + b_i,$$

where the symbol f demonstrates the sigmoid function;

- *output layer unit:*

$$Net(x) = \sum_{i=1}^{I} \left(w_i^2.o_i^2\right) = \sum_{i=1}^{I} \left(w_i^2 f\left(w_i^1.x + b_i\right)\right). \tag{7}$$

It seems necessary to mention at this point that after making some slight alterations to this pro type network model, it becomes an efficient tool for modeling the main problem.

3 Description of the proposed method

FOIDEs have gained a lot of attention due to their applicability in various scientific disciplines. Generally, there is no straightforward way to find exact solutions of fractional problems. As such, researchers must draw inferences from the suggested arbitrary numerical methods. The primary objective of this section is to use the ANN approach to find the approximate solution for a given class of fractional integro-differential equations (see Fig. 1). Hence, a preliminary value is considered for ordinary linear Volterra type integro-differential equations with fractional order requiring the Caputo type derivative of the form

$$P(x).{}_aD_x^{\alpha_1}\big[u(x)\big] + Q(x).I_{a,x}^{\alpha_2}\big[u(t)\big] = H(x), \quad 1 < \alpha_1, \alpha_2 \leq 2, a \leq x \leq b, \tag{8}$$

under the influence of initial conditions

$$u(a) = \beta_1 \quad \text{and} \quad u'(a) = \beta_2.$$

Here P, Q, and H are specified real-valued analytic functions on the continuous region (a, b). As is obvious, a power optimization technique is fundamentally comprised of a finite Maclaurin series approximation of a solution function with a successful optimization strategy. The power series (PS) method is applicable for estimating the solution of a minimization (or maximization) problem on a given region. From now on, we desire to use the PS method to approximate the unknown function $u(x)$, after rewriting it in an applicable trial solution form. This means that in order to employ this strategy, the initial conditions should be firstly applied to the origin problem. For the alluded equation the trial solution is written as follows:

$$\tilde{u}(x) = \beta_1 + \beta_2 x + x^2 \sum_{i=1}^{I} \big(w_i^2 . f\big(w_i^1 . x + b_i\big)\big). \tag{9}$$

During the process of this research, an attempt will be made to approximate the network parameter vectors w^1, w^2, and b, utilizing the most applicable BP machine learning algorithm.

3.1 Formulating a minimization problem

As shown in the previous part, the intended ANN architecture can be modeled completely and imitate the fractional problem (8) with the assistance of the trial solution (9). Here it is important to be aware of the fact that the neural network needs to be fully trained prior to treating it as an option for the unknown function $u(x)$. In this way, it must be pointed out that the learning objective means finding proper quantitative values for the parameters of the network, i.e., w_i^1, w_i^2, and b_i (for $i = 1, \ldots, n$), in such a way as to approximate the solution function with high precision. Therefore, the origin problem (8) is reduced to a corresponding minimization problem through discretizing the specific domain $\Omega = (a, b)$. In this discretization procedure, Ω_r is a partition of the domain Ω with the nodal points $x_r = a + \frac{r(b-a)}{R}$ (for $r = 0, \ldots, R$, $R \in \mathbf{N}$). For simplification, the research is continued under the supposition that $(a, b) = (0, 1)$. More generally, any case is capable of being absolutely

transformed to this circumstance by the linearization operator $\frac{x}{b-a} + \frac{a}{a-b}, x \in (a,b)$. Replacing the above trial solution in equation (8) contributes to the following applicable form:

$$P(x).{}_0D_x^{\alpha_1}\left[\beta_1 + \beta_2 x + x^2 \sum_{i=1}^{I}\left(w_i^2.f\left(w_i^1.x + b_i\right)\right)\right] \tag{10}$$
$$+ Q(x).I_{0,x}^{\alpha_2}\left[\beta_1 + \beta_2 t + t^2 \sum_{i=1}^{I}\left(w_i^2.f\left(w_i^1.t + b_i\right)\right)\right] = H(x), \quad x \in \Omega.$$

Here one must spread out the operators $D_x^{\alpha_1}$ and $I_{0,x}^{\alpha_2}$ in the series involving the non-linear activation function f. From a mathematical point of view, computing the fractional-order derivative and integral for the non-linear function f is very complicated. An alternative scheme must be found so as to assist us in explaining the issue. Now, for calculating higher-order fractional derivatives, a recurrence relation is proposed [16]:

$$f^{(n)}(x) = \sum_{k=1}^{n+1}(-1)^{k-1}\xi_k^n f^k, \tag{11}$$
$$\xi_k^n = (k-1)\xi_{k-1}^{n-1} + k\xi_k^{n-1},$$
$$\xi_k^n = 0, \quad n < 0, k < 1, k > n+1.$$

The constant coefficients ξ_k^n for the initial values of n and k are shown in Table 1. After replacing equation (11) in equation (10) and simplifying the result, the following result is obtained:

$$P(x).{}_0D_x^{\alpha_1}\left[x^2 \sum_{i=1}^{I}\left(w_i^2.\sum_{n=0}^{\infty}\frac{f^{(n)}(0)}{n!}\left(w_i^1.x + b_i\right)^n\right)\right] \tag{12}$$
$$+ Q(x).I_{0,x}^{\alpha_2}\left[t^2 \sum_{i=1}^{I}\left(w_i^2.\sum_{n=0}^{\infty}\frac{f^{(n)}(0)}{n!}\left(w_i^1.t + b_i\right)^n\right)\right]$$
$$+ Q(x).\frac{\beta_1}{\Gamma(1+\alpha_2)}.x^{\alpha_2} + Q(x).\frac{\beta_2}{\Gamma(2+\alpha_2)}.x^{1+\alpha_2} = H(x), \quad x \in \Omega.$$

Now, we have

$$P(x).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x^{j-\alpha_1+2}.(b_i)^{n-j} \tag{13}$$
$$+ Q(x).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x^{j+\alpha_2+2}.(b_i)^{n-j}$$
$$+ Q(x).\frac{\beta_1}{\Gamma(\alpha_2+1)}.x^{\alpha_2} + Q(x).\frac{\beta_2}{\Gamma(\alpha_2+2)}.x^{\alpha_2+1} = H(x), \quad x \in \Omega.$$

To complete this strategy, the identified points x_r (for $r = 0, \ldots, R$) are put into equation (13). In the end, the differentiable least mean square (LMS) algorithm is used to improve

Table 1 The constant coefficients ξ_k^n

n	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
0	1	0	0	0	0
1	1	1	0	0	0
2	1	3	2	0	0
3	1	7	12	6	0
4	1	15	50	60	24

the optimization strategy as follows:

$$\begin{aligned} E_r = \frac{1}{2}\Bigg(& P(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}.(b_i)^{n-j} \\ & + Q(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x_r^{j+\alpha_2+2}.(b_i)^{n-j} \\ & + Q(x_r).\frac{\beta_1}{\Gamma(\alpha_2+1)}.x_r^{\alpha_2} + Q(x_r).\frac{\beta_2}{\Gamma(\alpha_2+2)}.x_r^{\alpha_2+1} - H(x_r)\Bigg)^2, \quad r=0,\ldots,R. \end{aligned} \tag{14}$$

Improving this system using a suitable error rectification strategy is our objective in the following part. For more details, see reference [8].

3.1.1 Proposed machine learning approach

As reasonably explained above, in equation (8), the indicated integro-differential fractional initial value problem was meant to be converted into an optimization model through applying the prominent LMS rule. To find the solution of the obtained system, the network error needs to be significantly optimized on the reduced network parameter space (weights and biases). To do so, the quadratic error function, consisting of the sum of the squared network errors, E_r (for $r=0,\ldots,R$), is minimized benefiting from the standard BP (gradient descent-based) algorithm. To train a neural network, BP is an often used repetitive learning procedure on the foundation of the adjustment of weights and biases. At the beginning, the parameters of the network, i.e., w_i^1, w_i^2, and b_i, are real-valued random constants for training. Then, the differentiable function $E=\sum_{r=0}^{R}$ is improved via the suggested supervised BP learning rule. To do so, the algorithm is established for parameter w_i^2 as follows:

$$\begin{aligned} & w_i^2(\tau+1) = w_i^2(\tau) + \Delta w_i^2(\tau), \\ & \Delta w_i^2(\tau) = -\eta.\frac{\partial E}{\partial w_i^2} + \gamma.\Delta w_i^2(\tau-1), \quad i=1,\ldots,I, \\ & \frac{\partial E}{\partial w_i^2} = \sum_{r=0}^{R}\frac{\partial E_r}{\partial w_i^2}, \end{aligned} \tag{15}$$

where τ, η, and γ are the repetition step number, learning rate, and momentum term, respectively. Furthermore, the indexes $w_i^2(\tau+1)$ and $w_i^2(\tau)$ depict the adjusted and current weight parameter for each label of the training subscript i, respectively. In order to

complete the learning process, the partial derivative $\frac{\partial E_r}{\partial w_i^2}$ is given as follows:

$$\begin{aligned}\frac{\partial E_r}{\partial w_i^2} = &\left(P(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}.(b_i)^{n-j}\right.\\ &+ Q(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x^{j+\alpha_2+2}.(b_i)^{n-j}\\ &\left.+ Q(x_r).\frac{\beta_1}{\Gamma(\alpha_2+1)}.x_r^{\alpha_2} + Q(x_r).\frac{\beta_2}{\Gamma(\alpha_2+2)}.x_r^{\alpha_2+1} - H(x_r)\right)\\ &\times\left(\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.\left(w_i^1\right)^j.(b_i)^{n-j}\left(P(x_r).\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}\right.\right.\\ &\left.\left.+ Q(x_r).\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x^{j+\alpha_2+2}\right)\right).\end{aligned}$$

In a process similar to that for the weight parameter w_i^2, this modifying routine is reiterated for parameter w_i^1 as follows:

$$\begin{aligned}&w_i^1(\tau+1) = w_i^1(\tau) + \Delta w_i^1(\tau),\\ &\Delta w_i^1(\tau) = -\eta.\frac{\partial E}{\partial w_i^1} + \gamma.\Delta w_i^1(\tau-1),\end{aligned} \tag{16}$$

where

$$\begin{aligned}\frac{\partial E}{\partial w_i^1} = \sum_{r=0}^{R}\Bigg(&\left(P(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}.(b_i)^{n-j}\right.\\ &+ Q(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x^{j+\alpha_2+2}.(b_i)^{n-j}\\ &\left.+ Q(x_r).\frac{\beta_1}{\Gamma(\alpha_2+1)}.x_r^{\alpha_2} + Q(x_r).\frac{\beta_2}{\Gamma(\alpha_2+2)}.x_r^{\alpha_2+1} - H(x_r)\right)\\ &\times\left(\sum_{n=0}^{\infty}\sum_{j=0}^{n} C_n.\binom{n}{j}.w_i^2.j.\left(w_i^1\right)^{j-1}.(b_i)^{n-j}\left(P(x_r).\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}\right.\right.\\ &\left.\left.\left.+ Q(x_r).\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x_r^{j+\alpha_2+2}\right)\right)\Bigg).\end{aligned}$$

In this case, the bias parameter adjustment relations are identical to those in the example given above. Hence, we gain

$$\begin{aligned}&b_i(\tau+1) = b_i(\tau) + \Delta b_i(\tau),\\ &\Delta b_i(\tau) = -\eta.\frac{\partial E}{\partial b_i} + \gamma.\Delta b_i(\tau-1),\end{aligned} \tag{17}$$

where

$$\begin{aligned}\frac{\partial E}{\partial b_i} = \sum_{r=0}^{R}\Bigg(\Bigg(&P(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n}C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}.(b_i)^{n-j}\\&+Q(x_r).\sum_{i=1}^{I}\sum_{n=0}^{\infty}\sum_{j=0}^{n}C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x^{j+\alpha_2+2}.(b_i)^{n-j}\\&+Q(x_r).\frac{\beta_1}{\Gamma(\alpha_2+1)}.x_r^{\alpha_2}+Q(x_r).\frac{\beta_2}{\Gamma(\alpha_2+2)}.x_r^{\alpha_2+1}-H(x_r)\Bigg)\\&\times\Bigg(\sum_{n=0}^{\infty}\sum_{j=0}^{n}C_n.\binom{n}{j}.w_i^2.\left(w_i^1\right)^j.(n-j).(b_i)^{n-j-1}\Bigg(P(x_r).\frac{\Gamma(j+3)}{\Gamma(j-\alpha_1+3)}.x_r^{j-\alpha_1+2}\\&+Q(x_r).\frac{\Gamma(j+3)}{\Gamma(j+\alpha_2+3)}.x_r^{j+\alpha_2+2}\Bigg)\Bigg)\Bigg).\end{aligned}$$

To a large extent, an initial value fractional higher-order linear Volterra type integro-differential equation is carefully regarded as follows:

$$P(x).{}_aD_x^{\alpha_1}\left[u(x)\right]+Q(x).I_{a,x}^{\alpha_2}\left[u(t)\right]=H(x),\quad a\le x\le b, \tag{18}$$

with initial conditions

$$\begin{aligned}&u(a)=\beta_1,\\&u'(a)=\beta_2,\\&\vdots\\&u^{(m)}(a)=\beta_{m+1},\end{aligned}$$

where $m<\alpha_1\le m+1$, $m'<\alpha_2\le m'+1$, $m, m'\in\mathbf{N}^{>1}$. The investigated trial solution for this problem is chosen as follows:

$$\tilde{u}(x)=\sum_{i=0}^{m}\beta_{i+1}x^i+x^{m+1}N(x). \tag{19}$$

To keep up with the procedure, the trial solution (19) has been replaced with equation (18) and various simplifications have been performed. As a result, the parallel optimization system is fulfilled for $x=x_r$. As previously indicated, the resulting problem might be minimized with the help of the BP learning rule. Please be aware that to shun overstatement, the related updating relations are not rewritten here.

4 Illustrative examples

Two test sample problems are treated in this part to show the productivity and suitability of the suggested method. Based on the data gathered below, a comparison is made with a method described in [18] to contribute to a better understanding and to show the accuracy of the proposed method. The consecutive mathematical calculations have been carried out employing the analytic software Matlab-R2013b. Parameters were set as follows:

1. learning rate $\eta = 0.05$,
2. momentum constant $\gamma = 0.01$,
3. PS limitation $N = 6$,
4. number of nodal points $R = 11$.

Example 4.1 First, presume the following higher-order linear fractional Volterra type integro-differential equation:

$$D_x^{1.5}\big[u(x)\big] + I_{0,x}^2\big[u(t)\big] = \frac{\Gamma(3)}{\Gamma(\frac{1}{5})}x^{\frac{1}{2}} + \frac{\Gamma(3)}{\Gamma(5)}x^4, \quad 0 \le x \le 1,$$

with primary requirements $u(0) = 1$, $u'(0) = 0$ and the accurate solution $u(x) = x^2 + 1$. To proceed, it would be crucial to determine the network parameters w_i^1, w_i^2, and b_i (for $i = 1, \ldots, 5$) with real-valued random constants. The achieved modified data are then applied, and the net parameters are adjusted for $\tau = 1000$ in succession. The accessed results are illustrated in Table 2, confirming the accuracy of the technique introduced in this study. The demonstrated total network error E is plotted in Fig. 2. In addition, the accurate and proximate solutions are plotted in Fig. 3 for several numbers of repetitions. The absolute errors between the exact and the approximate solutions are shown in Fig. 4. The profi-

Table 2 Numerical outcomes for Example 4.1 (for $l = 2$)

x	Solution		Absolute error
	Exact	Approximate	
0.1	1.0100	1.010031652495624	3.1652×10^{-5}
0.2	1.0400	1.040050245132188	5.0245×10^{-5}
0.3	1.0900	1.090051494824863	5.1494×10^{-5}
0.4	1.1600	1.160037912535227	3.7912×10^{-5}
0.5	1.2500	1.250013755435422	1.3755×10^{-5}
0.6	1.3600	1.360009123128541	9.1231×10^{-6}
0.7	1.4900	1.490028884324511	2.8884×10^{-5}
0.8	1.6400	1.640038231158423	3.8231×10^{-5}
0.9	1.8100	1.810030512245391	3.0512×10^{-5}

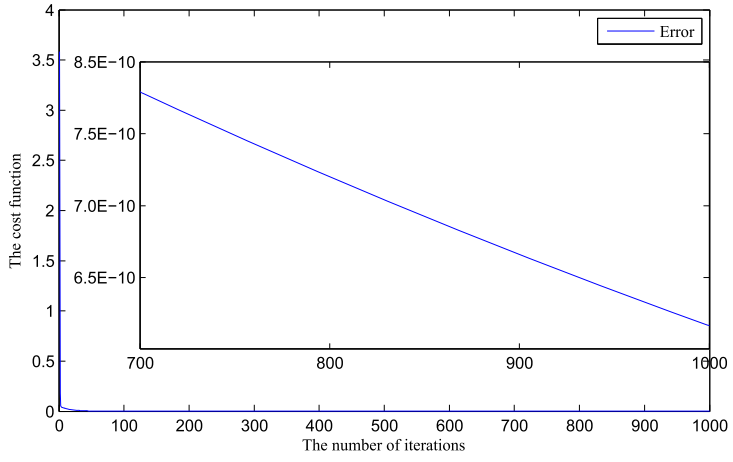

Figure 2 The error function for Example 4.1

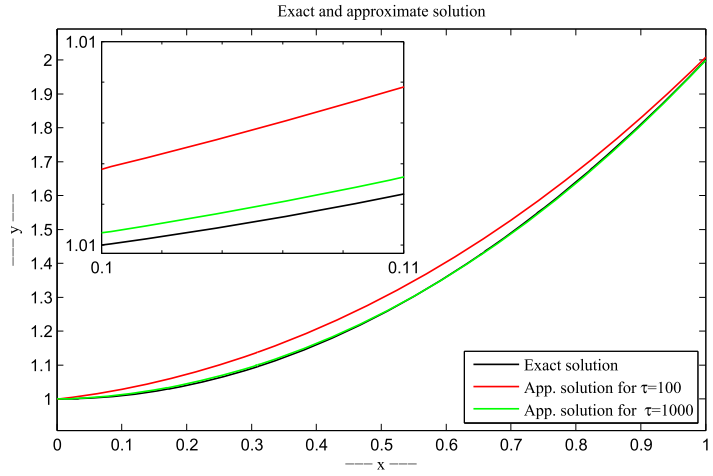

Figure 3 Exact and approximate solutions for Example 4.1

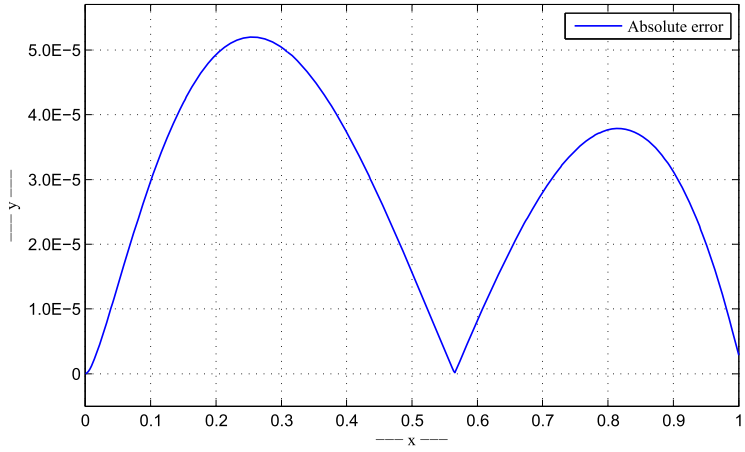

Figure 4 Absolute errors for Example 4.1

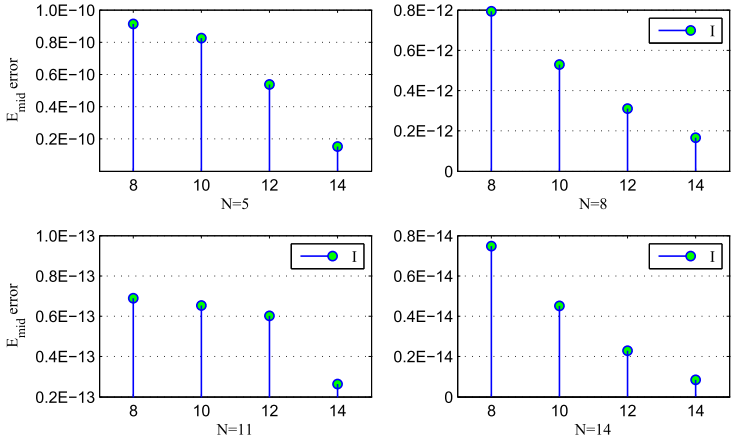

Figure 5 Suitability of the designed ANN architecture for Example 4.1

ciency of the designed ANN structure for varied control bases are illustrated using the E_{mid} function in Fig. 5.

Note that each time the training procedure was performed, the adjustable parameters were randomly selected as small positive real numbers.

Example 4.2 Consider the following fractional initial value problem:

$$D_x^2\big[u(x)\big] + D_x^{\frac{1}{2}}\big[u(x)\big] + u(x) - I_{0,x}^1\big[(x-t)u(t)\big] = R(x),$$
$$R(x) = -\frac{1}{12}x^4 + x^2 + \frac{2}{\Gamma(\frac{5}{2})}x^{\frac{3}{2}} + 2,$$

with initial conditions $u(0) = 1$, $u'(0) = 1$ and the exact solution $u(x) = x^2$ on a finite domain $0 \le x \le 1$. The main purpose of this example is to compare the numerical results (E_{mid} errors) obtained from the proposed model with the samples acquired by the Bessel

Table 3 E_{mid} errors for Example 4.2

x	ANN, $\tau = 1000$	ANN, $\tau = 10{,}000$	Method in [18]
	$l = 2$	$l = 2$	$N = 2$
0.0	0.2482×10^{-6}	0.7101×10^{-9}	0.5810×10^{-5}
0.1	0.8718×10^{-6}	0.1967×10^{-8}	0.9120×10^{-7}
0.2	0.1040×10^{-5}	0.3643×10^{-8}	0.5300×10^{-5}
0.3	0.8125×10^{-6}	0.9417×10^{-9}	0.1000×10^{-4}
0.4	0.7053×10^{-6}	0.1660×10^{-8}	0.1500×10^{-4}
0.5	0.6923×10^{-6}	0.2034×10^{-8}	0.2000×10^{-4}
0.5	0.5508×10^{-6}	0.8347×10^{-9}	0.2400×10^{-4}
0.7	0.4234×10^{-6}	0.9604×10^{-9}	0.2700×10^{-4}
0.8	0.4037×10^{-6}	0.2758×10^{-8}	0.3100×10^{-4}
0.9	0.3561×10^{-6}	0.2061×10^{-8}	0.3400×10^{-4}
1.0	0.2381×10^{-6}	0.8845×10^{-9}	0.3700×10^{-4}
x	ANN, $\tau = 1000$	ANN, $\tau = 10{,}000$	Method in [18]
	$l = 4$	$l = 4$	$N = 4$
0.0	0.9307×10^{-8}	0.9743×10^{-12}	0.2040×10^{-6}
0.1	0.1874×10^{-7}	0.2627×10^{-11}	0.1850×10^{-6}
0.2	0.3520×10^{-7}	0.3322×10^{-11}	0.6490×10^{-6}
0.3	0.6172×10^{-7}	0.9051×10^{-12}	0.1130×10^{-5}
0.4	0.2563×10^{-7}	0.2143×10^{-11}	0.1600×10^{-5}
0.5	0.1982×10^{-8}	0.1927×10^{-11}	0.2030×10^{-5}
0.5	0.8369×10^{-7}	0.8836×10^{-12}	0.2390×10^{-5}
0.7	0.6991×10^{-7}	0.1007×10^{-11}	0.2680×10^{-5}
0.8	0.4207×10^{-7}	0.3747×10^{-11}	0.2910×10^{-5}
0.9	0.2017×10^{-7}	0.2067×10^{-11}	0.3100×10^{-5}
1.0	0.9143×10^{-8}	0.1015×10^{-11}	0.3280×10^{-5}
x	ANN, $\tau = 1000$	ANN, $\tau = 10{,}000$	Method in [18]
	$l = 6$	$l = 6$	$N = 6$
0.0	0.2980×10^{-9}	0.8024×10^{-13}	0.5810×10^{-7}
0.1	0.7135×10^{-9}	0.9115×10^{-13}	0.9120×10^{-7}
0.2	0.1076×10^{-8}	0.2468×10^{-12}	0.5300×10^{-6}
0.3	0.3855×10^{-8}	0.4046×10^{-12}	0.1000×10^{-6}
0.4	0.1969×10^{-8}	0.7361×10^{-12}	0.1500×10^{-6}
0.5	0.8299×10^{-9}	0.3997×10^{-12}	0.2000×10^{-6}
0.5	0.8064×10^{-9}	0.1643×10^{-12}	0.2400×10^{-6}
0.7	0.6823×10^{-8}	0.9325×10^{-13}	0.2700×10^{-6}
0.8	0.4347×10^{-8}	0.8211×10^{-13}	0.3100×10^{-6}
0.9	0.1994×10^{-8}	0.6842×10^{-13}	0.3400×10^{-6}
1.0	0.9361×10^{-9}	0.5017×10^{-13}	0.3700×10^{-6}

polynomials method presented in [18], which are given in Table 3. These results allow us to claim that the proposed hybrid algorithm is able to approximate the unknown function with desired accuracy.

5 Conclusion

In this study, a combination of ANN and the PS approach has been effectually employed to approximate the solution of a Caputo type ordinary higher-order linear fractional Volterra integro-differential problem. To transmute the mentioned specific fractional problem into a minimization one, some felicitous features of the PS method together with the LMS rule were implemented. Not so long ago, ANNs' copious structures were definitely ingrained in the modeling and simulation of numerous realistic intricate phenomena. This ongoing work is vehemently dedicated to the investigation of a class of ordinary linear Volterra type integro-differential equations with fractional order in numerical mode. By replacing the unknown function by an appropriate multilayered feed-forward type neural structure, the fractional problem of such initial value is changed into a course of non-linear minimization equations, to some extent. Because of the exceedingly complex structure of the observed problem, the error BP algorithm was used by considering slight adjustments in the learning procedure. The designed multilayer neural architecture was then utilized to approximate the optimization problem on given sub-domains. Two fractional problems were tested to deal with the dependability of the present numerical method. Comparative comparison of the obtained numerical solution with corresponding exact ones for different partitionings of the solution domain revealed that the proposed technique is very effective and reliable. Providing fractional derivatives of different orders of the employed type of activation function in the hidden neurons is by far the most important result of this research. To achieve this, employing an effectual formulation was a definite must to calculate fractional derivatives of the sigmoidal function. This article is expected to underline the significance of the proposed method not only to solving iterative functions but also to other studies in related fields or specific areas. By expanding the recommended strategy to a broad class of non-linear situations, the shortcomings of prior research can be overcome and new ideas can be found to solve new problems.

Acknowledgements
The authors would like to thank the editor and the reviewers for the detailed and valuable suggestions that helped to improve the original manuscript to its present form.

Funding
The work of UFG was supported by the government of the Basque Country for the ELKARTEK21/10 KK-2021/00014 and ELKARTEK20/78 KK-2020/00114 research programs, respectively.

Availability of data and materials
Not applicable.

Declarations

Competing interests
The authors declare no competing interests.

Author contributions
TA is the supervisor of this study and was a major contributor in methodology, investigation and validation. AJ and RS worked on resources, investigation and formal analysis of this study. SMN, FK, UFG and SN worked on software, writing—review and editing and validation of the results. All authors have main contributions in writing the original draft preparation and also writing—review and editing the paper. All authors read and approved the final manuscript.

Author details
[1]Faculty of Engineering and Natural Sciences, Istinye University, Istanbul, Turkey. [2]Department of Mathematics, Urmia Branch, Islamic Azad University, Urmia, Iran. [3]Computer Engineering Department, Engineering Faculty, Fatih Sultan Mehmet Vakif University, Istanbul, Turkey. [4]Nuclear Engineering and Fluid Mechanics Department, University of the Basque Country UPV/EHU, Nieves Cano 12, 01006 Vitoria-Gasteiz, Spain. [5]Industrial Mathematics Laboratory, Baikal School of BRICS, Irkutsk National Research Technical University, Irkutsk, 664074, Russia. [6]Department of Applied Mathematics and Programming, South Ural State University, Lenin prospect 76, Chelyabinsk, 454080, Russia.

Received: 6 April 2022 Accepted: 2 July 2023 Published online: 18 July 2023

References

1. Alkan, S., Hatipoglu, V.: Approximate solutions of Volterra–Fredholm integro-differential equations of fractional order. Tbil. Math. J. **10**(2), 1–13 (2017)
2. Bazgir, H., Ghazanfari, B.: Spectral solution of fractional fourth order partial integro-differential equations. Comput. Methods Differ. Equ. **7**(2), 289–301 (2019)
3. Bentrcia, T., Mennouni, A.: On the asymptotic stability of a Bresse system with two fractional damping terms: theoretical and numerical analysis. Am. Inst. Math. Sci. **28**(1), 580–622 (2023)
4. Caputo, M.: Linear model of dissipation whose Q is almost frequency dependent II. Geophys. J. R. Astron. Soc. **13**, 529–539 (1967)
5. Elbeleze, A.A., Kilicman, A., Taib, B.M.: Approximate solution of integro-differential equation of fractional (arbitrary) order. J. King Saud Univ., Sci. **28**(1), 61–68 (2016)
6. Graupe, D.: Principles of Artificial Neural Networks, 2nd edn. World Scientific, Singapore (2007)
7. Hamoud, A.A., Ghadle, K.P., Atshan, S.H.: The approximate solutions of fractional integro-differential equations by using modified Adomian decomposition method. Khayyam J. Math. **5**(1), 21–39 (2019)
8. Hassoun, M.H.: Fundamentals of Artificial Neural Networks. MIT Press, Cambridge (1995)
9. Jafarian, A., Measoomy Nia, S., Abbasbandy, S.: Artificial neural networks based modeling for solving Volterra integral equations system. Appl. Soft Comput. **27**, 391–398 (2015)
10. Jafarian, A., Measoomy Nia, S., Jafari, R.: Solving fuzzy equations using neural nets with a new learning algorithm. J. Adv. Comput. Res. **3**(4), 33–45 (2012)
11. Jafarian, A., Mokhtarpour, M., Baleanu, D.: Artificial neural network approach for a class of fractional ordinary differential equation. Neural Comput. Appl. **28**(4), 765–773 (2017)
12. Jafariana, A., Measoomy Nia, S.: An application of ANNs on power series method for solving fractional Fredholm type integro-differential equations. Neural Parallel Sci. Comput. **24** (2016)
13. Jafariana, A., Measoomy Nia, S., Golmankhaneh, A.K., Baleanu, D.: Onartificial neural networks approach with new cost functions. Appl. Math. Comput. **339**(15), 546–555 (2018)
14. Kurkcu, O.K.: An evolutionary numerical method for solving nonlinear fractional Fredholm–Volterra–Hammerstein integro-differential-delay equations with a functional bound. Int. J. Comput. Math. **99**(11), 2159–2174 (2022)
15. Mennouni, A.: Improvement by projection for integro-differential equations. Math. Methods Appl. Sci. (2020). https://doi.org/10.1002/mma.6318
16. Minai, A.A., Williams, R.D.: On the derivatives of the sigmoid. Neural Netw. **6**, 845–853 (1993)
17. Nemati, S., Lima, P.M.: Numerical solution of nonlinear fractional integro-differential equations with weakly singular kernels via a modification of hat functions. Appl. Math. Comput. **327**(15), 79–92 (2018)
18. Ordokhani, Y., Dehestani, H.: Numerical solution of linear Fredholm–Volterra integro-differential equations of fractional order. World J. Model. Simul. **12**(3), 204–216 (2016)
19. Rahimkhania, P., Ordokhani, Y., Babolian, E.: A numerical scheme for solving nonlinear fractional Volterra integro-differential equations. Iran. J. Math. Sci. Inform. **13**(2), 111–132 (2018)
20. Ross, B.: Fractional Calculus and Its Applications. Proceedings of the International Conference Held at the University of New Haven. Springer, Berlin (1974)
21. Rostami, F., Jafarian, A.: A new artificial neural network structure for solving high-order linear fractional differential equations. Int. J. Comput. Math. **95**(3), 528–539 (2018)
22. Soltani, Z., Jafarian, A.: A new artificial neural networks approach for diagnosing diabetes disease type II. Int. J. Adv. Comput. Sci. Appl. **7**(6), 89–94 (2016)
23. Wang, J.: Numerical algorithm for two-dimensional nonlinear Volterra–Fredholm integral equations and fractional integro-differential equations (of Hammerstein and mixed types). Eng. Comput. **38**(9), 3548–3563 (2021)
24. Wang, Y., Zhu, L.: Solving nonlinear Volterra integro-differential equations of fractional order by using Euler wavelet method. Adv. Differ. Equ. (2017). https://doi.org/10.1186/s13662-017-1085-6
25. Wei, J., Tian, T.: Numerical solution of nonlinear Volterra integro-differential equations of fractional order by the reproducing kernel method. Appl. Math. Model. **39**, 4871–4876 (2015)
26. Yang, A.M., Han, Y., Mang, Y.Z.: On local fractional Volterra integro-differential equations in fractal steady heat, transfer. Therm. Sci. **20**, 789–793 (2016)
27. Zhou, D., Zhang, K., Ravey, A., Gao, F., Miraoui, A.: Parameter sensitivity analysis for fractional-order modeling of lithium-ion batteries. Energies **9**(3), 1–26 (2016)

Publisher's Note