Available online at http://ijim.srbiau.ac.ir/
Int. J. Industrial Mathematics (ISSN 2008-5621)
Vol. 15, No. 3, 2023 Article ID IJIM-817, 109 pages
DOI: http://dx.doi.org/10.30495/ijim.2023.15261.817
Research Article
Science and Research Branch (IAU)

Extended Aartificial Neural Networks Approach and Fractional Volterra Integro-Differential Equations

A. Jafarian *†, R. Saneifard ‡

Received Date: 2022-04-12 Revised Date: 2022-08-11 Accepted Date: 2022-08-13

Abstract

The current research intends to apply an extended version of artificial neural networks approach for solving a linear fractional Volterra integro-differential equation. More precisely, our main discussion in this study is to answer this fundamental question practically: That is, would it be possible to speed up the convergence of a given neural network structure by substituting the standard first-order derivative with a fractional order one. It is worth mentioning that, there is no certain geometric interpretation for fractional order derivative of a continuous function at a point in the domain. But, we strongly believe that this derivative calculus might help us to discover an efficient iterative technique to approximate solution of the earlier mentioned math problem. In this way, two simple but typical comparative test problems are given, and the related salient features of this technique are also discussed.

Keywords : Fractional Volterra integro-differential equation; Fractional artificial neural network; Least mean squares cost function; Supervised back-propagation learning algorithm.

1 Introduction

An overview on studies in the field of mathematics especially in the numerical analysis aspects reveals the fact that even after years of development, many available numerical packages are still not freely responsible for solving complicated mathematical problems. It is worth mentioning that many physical phenomena and issues related to engineering are modeled in terms of complex non-linear equation, and it is difficult to solve them analytically. The importance of obtaining solutions in modern mathematics problems has led to the fact that scientists are looking for most efficient alternative schemes. The recent developments in engineering science and technology rise to new branches of mathematics in which can be applied for modeling the related issues. Fractional problems are newly research topics in mathematical analysis, that many authors have paid attention to study their solutions. Recently extensive works have been done to solve fractional differential equations, due to their increasingly applications in engineering sciences and applied physics. In the last decades, a variety of powerful and effective methods such as

*Corresponding author. jafarian5594@yahoo.com, Tel:+98(914)1893041.
†Department of Mathematics, Urmia Branch, Islamic Azad University, Urmia, Iran.
‡Department of Mathematics, Urmia Branch, Islamic Azad University, Urmia, Iran.

Taylor expansion method [7], Hybrid collocation method [12], Chebyshev collocation method [14], CAS Wavelets method [21], differential transform method [15], He's variational iteration method [8], Homotopy analysis method [4], Euler wavelet method [18], SCW method [17], hybrid collocation method [13] and reproducing kernel method [19] have been proposed to the numerical solution of linear and non-linear fractional problems. A review of recently developed methods shows that the inbuilt deficiencies of these techniques are not letting the current conventional approaches to be useful for modeling and solving many different types of such problems. Inspired and motivated by the ongoing research in this area, we use an elegant modification of the artificial neural networks (ANNs) approach coupled with the power series method to investigate solution of a fundamental linear fractional Volterra integro-differential equation. In general, the indicated method is a non-algorithmic self-learning iterative technique which has been designed based on the performance of the human brain neuron system. Until now, several structures of ANNs have been employed to the numerical solution of fractional problems due to the complexity of mathematical problems derived from functional models [9, 16, 10]. It can be described in the frame work of the neural networks approach that the convergence control parameters are the main auxiliary tools which distinguish this technique from the other existing ones. In this paper, a three-layered feed-forward fractional type neural network is considered to the numerical solution of a linear fractional Volterra integro-differential equation. The main advantage of proposed structure over conventional ones is using fractional derivative instead of integer one in the training steps. The designed neural structure first, combine the given initial condition within the origin problem, and then the resulted neural net is substituted with the unknown function. Employing the least mean square algorithm leads to get a suitable criterion function. Now, the defined cost function converts the fractional differential equation to a non-linear optimization problem. Updating the unknown network parameters using a learning algorithm which is based on the fractional type gradient descent rule, reduces the network error on the training iterations. These iterations are repeated until the amount of network error tends to zero. The following strategy is organized below. Within this research framework, an overview on neural networks and their computational process in solving the mentioned differential equation is stated in section 2. In section 3, two test problems are solved through the presented ANNs approach, and comparison is made with the solutions obtained from the standard one. It has been shown numerically that the proposed ANN architecture presents a desirable convergence of the solution in compared with other method reported in this literature. At last, conclusions regarding the text and recommendations for future works are also pointed out in section 4.

2 Description of the method

The purpose of this section is to provide an accurate approximate solution for the linear fractional Volterra integro-differential equation problem. It is evident from discussions here, ANNs approach and power series method as two fundamental parts of numerical analysis, have relevant places in constructing algorithms for solving so many complicated math problems. The combination of these two famous methods has become much more efficient tool in solving a large part of mathematical problems. The main reason consists in the fact that these both have their unique advantages, which led to their increasing application in approximating solution of complex problems. Now, the notable details in neural networks are summarized here.

Artificial neural networks facilitate solving the various problems in science. In this role, one of their major advantages is likely to be able to manage the process of solving wide range of problems as a simplistic model of real neural systems. They also provide researchers with greater access to the desired solutions. At the same time, as the neural networks make possible greater applications on various problems, they

need to be trained and we are going to be familiar with their capabilities as well. In this regard, there are also many works that researchers can draw on. But in majority of these works, the elements and essential principles of these networks are well established in details and it should be acknowledged that they certainly seem to be time consuming. In this part, we will try to determine fundamentals of modeling and analyzing fractional equation using neural networks. Still more, we would give brief required mathematical equations, some implementing tips, practical skills and techniques necessary engaged in this process. Although there is a breadth and depth of material available, here we attempt to avoid addressing unnecessary details which don't deserve a place in this paper. These days, of course, utilizing of intelligent systems, namely artificial neural networks is revived to some point that such tools can be categorized in both basic and common tools. You may have noticed that neural networks in most academic majors, step forward in dealing with some tasks such as: analyzing, decision-making, approximating, forecasting, designing and establishing. In what above, a brief idea about artificial neural networks has been presented, where details can be found in many references such as [3, 5].

To illustrate the procedure, we presented Figure 1 where it is a three-layer feed-forward neural network comprised of one input signal, I hidden neurons and one output neuron. As can be seen, information which moves into network through the input neurons are meant to be combined with the weight values corresponding to the input layer neurons. This procedure establishes the inputs to the hidden layer. The weights are in fact important to the inputs in the network. Activation function for the hidden layer is being provided to the input data. The activation function is defined by the verity of neural network types. The corresponding weight parameters to the third layer are then multiplied by outputs to the hidden layer and then do establish the output for the network. According to the above mentioned descriptions, input-output relations in each network layer can be summarized as follows:

- *Input unit:*

$$o_1^1 = x, \tag{2.1}$$

- *Hidden units:*

$$o_i^2 = f(net_i),\ i = 1, ..., I, \tag{2.2}$$

$$net_i = x.w_i^1,$$

where f and w^1 are activation function and hidden layer weight vector, respectively.

- *Output unit:*

$$N(x) = \sum_{i=1}^{I}(w_i^2.o_i^2) + b = \sum_{i=1}^{I}(w_i^2.f(w_i^1.x)) + b, \tag{2.3}$$

where b and w^2 are bias term and output layer weight vector, respectively.

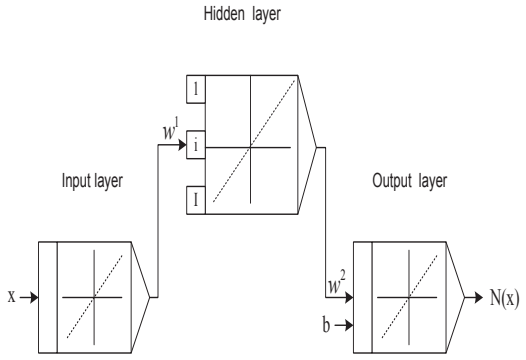

Figure 1: The designed neural network architecture

In the following parts, it will be shown that how the mentioned fractional problem can be solved by this simplified mathematical model of brain-like computational system.

2.1 Implementation of the method

Fractional calculus play a dominant role in the mathematical modeling of many phenomena related to physics and engineering sciences. Therefore, these fractional equations have attracted great attention, and investigation of new numerical techniques for solving these problems have become a fascinating task for many experts. Thus,

we would like here to extend the applications of both ANNs approach and power series method to seek the solution of linear fractional Volterra integro-differential equations problem. This part of the study begins with the basic concepts of fractional calculations. As is evident from the references [2, 11], there are several definitions for fractional order derivative and integration. Here, we tend to choose the Caputo derivative definition to continue the computational procedure.

Definition 2.1. *Let $u(x)$ be 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 by:*

$$ {}_aD_x^\alpha[u(x)] = \begin{cases} \frac{d^k u(x)}{dx^k} & ,\alpha = k \in N \\ \frac{1}{\Gamma(k-\alpha)} \int_a^x \frac{u^{(k)}(\tau)}{(x-\tau)^{\alpha-k+1}} d\tau, & \\ x > a, 0 \le k-1 < \alpha < k & \end{cases}, \tag{2.4} $$

$$ I_{a,x}^\alpha[u(x)] = \frac{1}{\Gamma(\alpha)} \int_a^x \frac{u(\tau)}{(x-\tau)^{1-\alpha}} d\tau, \tag{2.5} $$

respectively. Here, $\Gamma(.)$ symbolizes the Gamma function. It must be mentioned that the derivative of a constant for the Caputo sense is zero, and also the following practical property holds:

$$ {}_aD_x^\alpha[x^k] = \begin{cases} 0 & k \in Z^+,\ k < \lceil \alpha \rceil \\ \frac{\Gamma(k+1)}{\Gamma(k+1-\alpha)} x^{k-\alpha}, & \\ x > ak \in Z^+,\ k \ge \lceil \alpha \rceil & \end{cases}, \tag{2.6} $$

$$ I_{0,x}^\alpha[t^k] = \frac{\Gamma(k+1)}{\Gamma(k+1+\alpha)} x^{k+\alpha},\ k \in Z^+. \tag{2.7} $$

In the above relations $\lceil \alpha \rceil$ indicates the smallest integer greater than or equal to constant α. For more information, please review [1, 20]. For the purpose of applications illustration of the designed multi-layered feed-forward neural architecture, we consider here the following linear fractional Volterra differential equation:

$$ P(x).{}_aD_x^{\alpha_1}[u(x)] + Q(x).I_{a,x}^{\alpha_2}[x^{l_1}t^{l_2}u(t)] = H(x), \tag{2.8} $$

$$ 0 < \alpha_1, \alpha_2 \le 1, $$

for the length interval (a,b) and subject to initial condition

$$ u(a) = u_0. $$

In the above differential equation problem, the variable coefficients P, Q and H are deemed continuous real-valued functions. In the last decade, extensive studies have been done on the use of power series method in solving several types of fractional differential equations. As previously mentioned, the designed neural architecture is considered as an iterative method, supposedly that the solution function $u(x)$ on domain $\Omega = [a,b]$, is completely represented in a power series of degree I. Let:

$$ u_I(x) = \sum_{i=0}^{I} a_i x^i, \tag{2.9} $$

be the referred generalized power series for constant coefficients a_i (for $i = 0, ..., I$). It should be noted here that the solution function $u(x)$ is represented as the series polynomial (2.9) if and only if it is complex differentiable in the open set (a,b). For simplicity reasons, this differential interval is replaced with the suitable domain $[0,T]$, which will be used throughout the paper. The relationship between designed neural network and indicated power series will be explained further in the following parts.

2.2 Discretization of the problems

As was seen in the previous part, the defined fractional problem consists of an initial condition, which must be incorporated in the framework of methodology. We now have to replace the solution function $u(x)$ with a more simplified one, that includes the initial condition. For the above mentioned problem, an initial trial particular solution which is used in the remaining parts of this paper is written in the form:

$$ \tilde{u}(x) = u_0 + x.N(x) = u_0 + x.(\sum_{i=1}^{I} w_i^2.f(w_i^1.x) + b). \tag{2.10} $$

Here, $\tilde{u}(x)$ involves the given feed-forward architecture that is satisfied in the initial condition, simultaneously. Here, the notations $w_i^2 = a_i$,

$w_i^1 = x^{i-1}$ and $b = a_0$ are used for corresponding the designed neural network within the power series (2.9). Substituting $u(x)$ with $\tilde{u}(x)$ leads to the following relation:

$$P(x).{}_0D_x^{\alpha_1}[u_0 + \sum_{i=0}^{I} a_i x^{i+1}] + \tag{2.11}$$

$$Q(x).I_{0,x}^{\alpha_2}[u_0.x^{l_1} + x^{l_1}.\sum_{i=0}^{I} a_i t^{i+l_2+1})] = H(x),$$

$$0 < x < T, 0 < \alpha_1, \alpha_2 \leq 1.$$

Expanding the defined fractional derivative ${}_0D_x^{\alpha_1}$ and fractional integration $I_{0,x}^{\alpha_2}$ on the series including designed neural architecture and then making some algebraic simplification, leads to the following relation:

$$P(x).\sum_{i=0}^{I} \frac{a_i.\Gamma(i+2).x^{i+1-\alpha_1}}{\Gamma(i+2-\alpha_1)} + \tag{2.12}$$

$$Q(x).\sum_{i=0}^{I} \frac{a_i.\Gamma(i+l_2+2).x^{i+l_1+l_2+1+\alpha_2}}{\Gamma(i+l_2+2+\alpha_2)}) +$$

$$Q(x).\frac{u_0.x^{l_1+\alpha_2}}{\Gamma(1+\alpha_2)} = H(x),\ 0 < x < T,\ 0 < \alpha_1, \alpha_2 \leq 1.$$

Now, a set of acceptable node points should be considered to complete the discretization procedure. For positive integer R, let Ω_r be a partition of domain Ω with the nodal points $x_r = \frac{rT}{R}$, (for $r = 0, ..., R$). Putting $x = x_r$ and then employing the differentiable least mean square (LMS) function reduces the resulted equation (2.12) into a non-linear optimization problem defined by the following system of equalities:

$$E_r = \frac{1}{2}\big(P(x_r).\sum_{i=0}^{I} \frac{a_i.\Gamma(i+2).x_r^{i+1-\alpha_1}}{\Gamma(i+2-\alpha_1)} + \tag{2.13}$$

$$Q(x_r).\sum_{i=0}^{I} \frac{a_i.\Gamma(i+l_2+2).x_r^{i+l_1+l_2+1+\alpha_2}}{\Gamma(i+l_2+2+\alpha_2)}) +$$

$$Q(x_r).\frac{u_0.x_r^{l_1+\alpha_2}}{\Gamma(1+\alpha_2)} - H(x_r))^2,$$

for $0 < \alpha_1, \alpha_2 \leq 1$ and $r = 0, ..., R$. Minimizing this cost function by means of the ANNs approach over the space of possible weight parameters is the main objective of the following part. Hence, one suitable error correction learning procedure must be essentially employed to fulfil this goal. More details concerning minimizing techniques can be found in [6].

2.3 Proposed learning algorithm

As a matter of fact, artificial neural networks are generally suitable to be employed as parallel distributed computing ones. They are processing such brain like systems to be shown that they are successful simplified mathematical models. The most characteristic of neural networks is that they need to be taught and trained. What it simply means is that they are being able to learn new compositions, new associations as well as new functional dependencies. On the other hand, experience shows that computers are being programmed to outperform specific issues relevant to biological and artificial neural systems. In either case, they are secured to keep the reliability, preciseness and speed arithmetic expressions. Along the same lines, these networks are constructed in a way that they appear to generate solid information as a new brand processing networks. Learning in neural networks is not that much straightforward. The network ought to run the desired computation. As such, it is required to find a set of connection strengths (weights). Obviously, a training set of example input-output pairs are given to the network. Due to approximating the function from which input-output pairs have been drawn, connections need to be modified. Then, the networks are invited to be examined to see if they are capable of generalizing. In remaining part of this section, we summarize the performance of proposed neural architecture. Still, the process of error correction learning is specifically provided simple enough. Here is the procedure: consider network is being training: now, input moves into the network and it is going to be a set of values on the output units. Next, wishing more proficient computation, actual output needs to be compared with the desired target. Now, if the output and target do match, we do not need to make any changes to the connections. No doubt, if the output differs from the

target, we have to take a step and make some changes. Regarding to this fact that convergence control parameters must be updated, the supervised back-propagation learning algorithm is employed to adjust the initial parameter a_i which has been chosen randomly. The mentioned algorithm is well presented as follows:

$$a_i(\tau+1) = a_i(\tau) + \Delta a_i(\tau),\ i = 0, ..., I, \quad (2.14)$$

$$\Delta a_i(\tau) = -\eta.D^{\beta}_{a_i}[E_r] + \gamma.\Delta a_i(\tau-1),\ 0 < \beta \leq 1,$$

where η, γ and β are the learning rate, momentum term and fractional order derivative, respectively. In the above relation, the indexes τ and i in $a_i(\tau)$ ascribe the iteration number and the label of training connection weight, respectively. Moreover, $a_i(\tau+1)$ and $a_i(\tau)$ depict the adjusted and current weight parameter, respectively. It is true that there is no certain geometric interpretation for fractional derivative of a continuous function at a given point of its domain, but we tend to do this to increase the degree of freedom in the neural network training. Practically, our task is to illustrate that there could be such order to the fractional derivative which by applying it in the procedure, network training will not be time consuming any more and there is going to be less error as well. Here, the fractional order generalized delta learning algorithm which is based on the gradient descent method is used to aid this training process. To complete the derivation of learning procedure, the above fractional order derivative is employed as follows:

$$D^{\beta}_{a_i}[E_r] = \frac{\Gamma(3)}{2.\Gamma(3-\beta)}\Big(P(x_r).$$

$$\sum_{i=0}^{I} \frac{a_i.\Gamma(i+2).x_r^{i+1-\alpha_1}}{\Gamma(i+2-\alpha_1)} - H(x_r) +$$

$$Q(x_r).\sum_{i=0}^{I} \frac{a_i.\Gamma(i+l_2+2).x_r^{i+l_1+l_2+1+\alpha_2}}{\Gamma(i+l_2+2+\alpha_2)}\Big) +$$

$$Q(x_r).\frac{u_0.x_r^{l_1+\alpha_2}}{\Gamma(1+\alpha_2)}\Big)^{2-\beta} \times$$

$$\Big(P(x_r).\frac{\Gamma(i+2).x_r^{i+1-\alpha_1}.a_i^{1-\beta}}{\Gamma(2-\beta).\Gamma(i+2-\alpha_1)} +$$

$$Q(x_r).\frac{\Gamma(i+l_2+2).x_r^{i+l_1+l_2+1+\alpha_2}.a_i^{1-\beta}}{\Gamma(2-\beta).\Gamma(i+l_2+2+\alpha_2)}\Big),$$

(for $r = 0, ..., R$ and $i = 0, ..., I$). With a little care in the above computational relations, it can easily be concluded that modifying the network parameters without the use of a powerful computational software is impossible. The symbolic computation software Matlab $v7.10$ is a high level one which researchers can employ it to omit wasting time and enhance the accuracy of calculations.

3 Illustrative examples

In this section, two fractional Volterra integro-differential equations problems are solved numerically to illustrate the capability of the developed ANNS approach. Besides, comparisons are conducted between the results of this iterative approach and those obtained by other numerical techniques proposed in [11, 12]. Here, the mean absolute error criterion $E_{mid} = \frac{1}{R+1}\sum_{r=0}^{R}|u(x_r) - \tilde{u}(x_r)|$ is implemented as an index to peruse the overall performance of each method. In the following simulations that illustrate the method discussed in detail, we use the specifications and standards as follows:

1. Learning rate $\eta = 0.01$,
2. Momentum constant $\gamma = 0.02$,
3. Number of hidden neurons $I = 6$,
4. Number of nodal points $R = 11$.

Example 3.1. *Consider the following linear fractional order integro-differential equation of Volterra type:*

$$D^{0.75}_{x}[u(x)] + I^{0.5}_{0,x}[u(t)] =$$

$$\frac{\Gamma(3)}{\Gamma(\frac{7}{2})}x^{\frac{5}{2}} + \frac{\Gamma(3)}{\Gamma(\frac{9}{4})}x^{\frac{5}{4}} + \frac{\Gamma(1)}{\Gamma(\frac{3}{2})}x^{\frac{1}{2}},\ 0 \leq x \leq 1,$$

with initial condition $u(0) = 1$ and the exact solution $u(x) = x^2 + 1$. The goal here is to apply the concepts learned in the previous section to the simulation of above fractional problem. At first, the weight parameters are quantified with

Table 1: Numerical results for Example 3.2

x	$\beta = \frac{\pi}{8}$	$\beta = \frac{\pi}{6}$	$\beta = \frac{\pi}{4}$	$\beta = 1$
	LMS error	*LMS error*	*LMS error*	*LMS error*
0.1	3.0188×10^{-9}	3.1278×10^{-7}	4.4283×10^{-13}	6.6602×10^{-10}
0.2	9.9716×10^{-8}	3.5484×10^{-7}	3.4472×10^{-13}	3.5518×10^{-10}
0.3	9.1028×10^{-8}	4.8918×10^{-7}	3.3180×10^{-13}	1.9684×10^{-10}
0.4	9.9430×10^{-8}	5.1376×10^{-7}	8.8143×10^{-14}	8.8308×10^{-10}
0.5	7.0088×10^{-8}	5.5627×10^{-7}	5.5390×10^{-13}	1.9479×10^{-09}
0.6	6.3176×10^{-8}	5.8328×10^{-7}	5.6903×10^{-13}	9.8321×10^{-10}
0.7	8.4976×10^{-8}	6.3275×10^{-7}	6.5140×10^{-13}	5.9641×10^{-10}
0.8	8.0873×10^{-8}	6.4822×10^{-7}	6.6517×10^{-13}	6.5577×10^{-10}
0.9	9.0113×10^{-8}	6.7391×10^{-7}	7.1820×10^{-13}	4.9429×10^{-10}

small real valued random constants. Then, the given differential interval $[0, 1]$ is partitioned with the help of regular discretization technique for $R = 11$. Now, the described learning algorithm (2.14) is implemented on the given training patterns to successively adjust the weight parameters until a suitable solution was found. Comparison between the LMS errors is made in Table 1, for $\tau = 500$. To demonstrate the accuracy of technique presented in this study, the indicated error functions are plotted in Figure 1, for different values of β. Moreover, the performance of proposed neural structure is perused using the E_{mid} error function in Figure 3, for different control elements ($\tau = 100$).

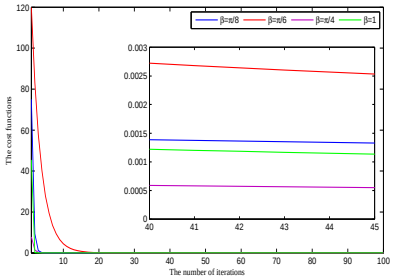

Figure 2: The cost functions for Example 3.1.

Example 3.2. *The following linear fractional Volterra integro-differential is considered:*

$$D_x^{0.5}[u(x)] - I_{0,x}^{1}[xtu(t)] = R(x),$$
$$u(0) = 1, \ 0 \le x \le 1,$$

where

$$R(x) = -\frac{\Gamma(\frac{5}{2})}{\Gamma(2)}x - \frac{x^3}{2} + (\frac{\Gamma(5)}{\Gamma(\frac{9}{2})} + \frac{2}{7})x^{\frac{7}{2}} - \frac{x^9}{6},$$

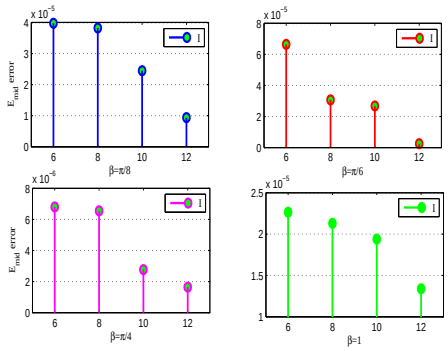

Figure 3: Performance of proposed neural architecture for Example 3.2.

with the exact solution $u(x) = 1 - x^{\frac{3}{2}} + x^4$. Similarly, the present iterative is employed for estimating solution of this fractional problem on domain $\Omega = [0, 1]$. Here, the obtained numerical results are compare with those obtained by our recent study proposed in [10] and the ones obtained from the hybrid collocation (HC) method [13]. The obtained least mean square errors are presented in Table 2, for $\tau = 1000$. We are now allowed to claim that, increasing the number of learning steps in the back-propagation algorithm leads to the accurately approximation of the unknown function.

All this makes it is easy to conclude that there exists at least a real number β belong to the interval $(0, 1)$, such that the neural network convergence speed is much more than conventional case studies.

Table 2: Comparison of approximate solutions for Example **??**

x	*ANNs approach*	*Method in* [10]	*Method in* [13]
	$\beta=\frac{e}{3},\ I=6$	$\beta=1,\ I=6$	$I=6$
0.1	6.8166×10^{-14}	6.9482×10^{-11}	4.7585×10^{-09}
0.2	7.0081×10^{-14}	9.1718×10^{-12}	5.5395×10^{-09}
0.3	7.3800×10^{-14}	9.5028×10^{-12}	5.6080×10^{-09}
0.4	7.6423×10^{-14}	3.3447×10^{-11}	7.7917×10^{-09}
0.5	7.9318×10^{-14}	4.3875×10^{-11}	8.3401×10^{-09}
0.6	8.0207×10^{-14}	3.8157×10^{-11}	1.2991×10^{-08}
0.7	8.1138×10^{-14}	7.9555×10^{-11}	2.6882×10^{-08}
0.8	1.0083×10^{-13}	7.5924×10^{-11}	9.1939×10^{-09}
0.9	1.0083×10^{-13}	1.8682×10^{-11}	9.6190×10^{-09}
x	$\beta=\frac{e}{3},\ I=8$	$\beta=1,\ I=8$	$I=8$
0.1	9.2186×10^{-16}	3.8975×10^{-14}	8.1428×10^{-11}
0.2	1.0067×10^{-15}	7.4554×10^{-13}	8.4352×10^{-11}
0.3	1.2934×10^{-15}	7.4632×10^{-13}	9.2926×10^{-11}
0.4	1.8533×10^{-15}	6.0937×10^{-14}	3.4998×10^{-10}
0.5	2.1160×10^{-15}	2.5468×10^{-14}	1.9660×10^{-10}
0.6	2.3784×10^{-15}	1.7607×10^{-14}	2.5108×10^{-10}
0.7	2.6931×10^{-15}	5.7975×10^{-14}	9.1604×10^{-11}
0.8	2.8890×10^{-15}	5.5514×10^{-14}	3.7329×10^{-10}
0.9	3.2177×10^{-15}	2.6262×10^{-14}	4.5166×10^{-10}
x	$\beta=\frac{e}{3},\ I=10$	$\beta=1,\ I=10$	$I=10$
0.1	3.2486×10^{-17}	8.1908×10^{-15}	8.3083×10^{-13}
0.2	3.8941×10^{-17}	3.9874×10^{-16}	2.8526×10^{-12}
0.3	4.3256×10^{-17}	2.5972×10^{-16}	5.4972×10^{-12}
0.4	4.7233×10^{-17}	2.4037×10^{-16}	9.1719×10^{-12}
0.5	5.0176×10^{-17}	4.8525×10^{-16}	2.8584×10^{-13}
0.6	5.4923×10^{-17}	1.2384×10^{-16}	7.5720×10^{-12}
0.7	5.9141×10^{-17}	6.5123×10^{-16}	7.5373×10^{-12}
0.8	6.2137×10^{-17}	1.5517×10^{-16}	7.8045×10^{-12}
0.9	6.5160×10^{-17}	4.0595×10^{-16}	8.6782×10^{-12}
x	$\beta=\frac{e}{3},\ I=12$	$\beta=1,\ I=12$	$I=12$
0.1	7.3591×10^{-18}	5.9904×10^{-17}	3.3712×10^{-14}
0.2	7.7083×10^{-18}	7.9095×10^{-17}	7.6218×10^{-13}
0.3	7.8996×10^{-18}	8.5927×10^{-16}	3.9428×10^{-13}
0.4	8.2760×10^{-18}	4.4726×10^{-17}	1.1122×10^{-13}
0.5	8.5173×10^{-18}	2.3868×10^{-17}	5.2853×10^{-13}
0.6	8.7916×10^{-18}	2.4928×10^{-17}	1.6565×10^{-13}
0.7	8.8993×10^{-18}	1.5756×10^{-17}	3.0198×10^{-13}
0.8	9.2170×10^{-18}	7.4075×10^{-17}	6.6297×10^{-13}
0.9	9.3756×10^{-18}	1.5424×10^{-17}	6.5408×10^{-13}

4 Conclusion

In the present work, we have applied a modification of the artificial neural networks approach to compute solution of a linear fractional Volterra integro-differential equation. In this new modification, instead of employing the usual first order derivative, a fractional order one has been applied in back-propagation learning process. In other words, the remarkable property of this technique is to use fractional displacement derivative with conventional first order one in the gradient descent rule. It has been demonstrated practically that the presented ANNs approach can be ap-

plied advantageously to approximate the power series solution of the given math problem. It is worth noting that, this method provides the solution in terms of convergent power series with easily computable components. We numerically showed that, in the case of fractional ANNs approach when the iteration steps become increasingly large, the convergence of the solution is very fast, and also the solution however becomes even more accurate. The two numerical examples illustratively analyzed the ability of modified ANNs approach presented in this study, and reveals the fact that this one case is very effective. Extensions of this technique can also be applied to higher order and non-linear situations.

References

[1] R. P. Agarwal, V. Lupulescu, D. ORegan, G. Rahman, Fractional calculus and fractional differential equations in nonreflexive Banach spaces, *Commun. Nonlinear Sci. Numer. Simul.* 20 (2015) 59-73.

[2] D. Baleanu, K. Diethelm, E. Scalas, J. J. Trujillo, Fractional calculus models and numerical methods, in: Series on Complexity, *Nonlinearity and Chaos, World Scientific, Boston*, (2012).

[3] R. Fuller, Neural fuzzy systems, Abo Akademi University press, (2005).

[4] I. K. Hanan, The Homotopy analysis method for solving multi-fractional order integro-differential equations, *Journal of Al-Nahrain University* 14 (2011) 139-143.

[5] M. Hanss, Applied Fuzzy Arithmetic: An introduction with engineering applications, *Springer-Verlag, Berlin*, (2005).

[6] M. H. Hassoun, Fundamentals of artificial neural networks, *MIT Press*, 1995.

[7] L. Huanga, X. F. Li, Y. Zhaoa, X. Y. Duana, Approximate solution of fractional integro-differential equations by Taylor expansion method, *Computers and Mathematics with Applications* 62 (2011) 1127-1134.

[8] S. Irandoust-Pakchin, S. Abdi-Mazraeh, Exact solutions for some of the fractional integro-differential equations with the nonlocal boundary conditions by using them modification of He's variational iteration method, *International Journal of Advanced Mathematical Sciences* 1 (2013) 139-144.

[9] A. Jafarian, M. Mokhtarpour, D. Baleanu, Artificial neural network approach for a class of fractional ordinary differential equation, *Neural Computing and Applications* http://dx.doi.org/10.1007/s00521-015-2104-8/.

[10] A. Jafarian, F. Rostami, A. K. Golmankhaneh, D. Baleanu, Using ANNs approach for solving fractional order Volterra integro-differential equations, *International Journal of Computational Intelligence Systems* 10 (2017) 470-480.

[11] V. Lakshmikantham, S. Leela, J. Vasundhara Devi, Theory of fractional dynamic systems, *Cambridge Univ. Press, Cambridge*, (2009).

[12] X. Ma, Ch. Huang, Numerical solution of fractional integro-differential equations by a hybrid collocation method, *Applied Mathematics and Computation* 219 (2013) 6750-6760.

[13] X. H. Ma, C. M. Huang, Numerical solution of fractional integro-differential equations by a hybrid collocation method, *Appl. Math. Comput.* 219 (2013) 6750-6760.

[14] D. Sh. Mohammed, Numerical solution of fractional integro-Differential equations by least squares method and shifted chebyshev polynomial, *Mathematical Problems in Engineering* http://dx.doi.org/10.1155/2014/431965/.

[15] D. Nazari, S. Shahmorad, Application of fractional differential transform method to the fractional order integro-differential equations with nonlocal boundary condition, *J. Comput. Appl. Math.* http://dx.doi.org/10.1016/j.cam.2010.01.053/

[16] F. Rostami, A. Jafarian, A new artificial neural network structure for solveing high-order linear fractional differential equations, *International Journal of Computer Mathematics* , http://dx.doi.org/10.1080/00207160.2017.1291932/.

[17] Y. X. Wang, L. Zhu, SCW method for solving the fractional integro-differential equations with a weakly singular kernel, *Appl. Math. Comput.* 275 (2016) 72-80 (2016).

[18] Y. Wang, L. Zhu, Solving nonlinear Volterra integro-differential equations of fractional order by using Euler wavelet method, *Advances in Difference Equations* , http://dx.doi.org/0.1186/s13662-017-1085-6/.

[19] J. Wei, T. Tian, Numerical solution of nonlinear Volterra integro-differential equations of fractional order by the reproducing kernel method, *Appl. Math. Model.* 39 (2015) 4871-4876 (2015).

[20] X. J. Yang, Advanced local fractional calculus and its applications, *World Science Publisher, New York, USA*, 2012.

[21] M. Yi, K. Sun, J. Huang, L. Wang, Numerical solutions of fractional integro-differential equations of Bratu type by using CAS wavelets, *Journal of Applied Mathematics*, http://dx.doi.org/10.1155/2013/801395/.

Ahmad Jafarian was born in 1978 in Tehran, Iran. He received B. Sc (1997) in Applied mathematics and M.Sc. in applied mathematics from Islamic Azad University Lahijan Branh to Lahijan, Ferdowsi University of Mashhad respectively. He is an Associate Prof. in the department of mathematics at Islamic Azad University, Urmia Branch, Urmia, in Iran. His current interest is in artificial intelligence, solving nonlinear problem and fuzzy mathematics.

Rahim Saneifard was born in 1972 in Oroumieh, Iran. He received Ph.D (2011) in applied mathematics from Since and Research Branch University to Tehran. He is a Associate Prof. in the department of mathematics at Islamic Azad University, Urmia Branch, Oroumieh, in Iran. His current interest is in fuzzy mathematics.