Applied Mathematics and Computation 293 (2017) 81–95

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Solving differential equations of fractional order using an optimization technique based on training artificial neural network

M. Pakdaman[a], A. Ahmadian[b], S. Effati[c], S. Salahshour[d], D. Baleanu[e,f,*]

[a] Young Researchers and Elite Club, Mashhad Branch, Islamic Azad University, Mashhad, Iran
[b] Department of Mathematics, Faculty of Science, University Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia
[c] Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran
[d] Young Researchers and Elite Club, Mobarakeh Branch, Islamic Azad University, Mobarakeh, Iran
[e] Department of Mathematics, Cankaya University, 06530 Balgat, Ankara, Turkey
[f] Institute of Space Sciences, Magurele-Bucharest, Romania

ARTICLE INFO

Keywords:
Multi-term fractional differential equations
Artificial neural network
Optimization
Caputo derivative

ABSTRACT

The current study aims to approximate the solution of fractional differential equations (FDEs) by using the fundamental properties of artificial neural networks (ANNs) for function approximation. In the first step, we derive an approximate solution of fractional differential equation (FDE) by using ANNs. In the second step, an optimization approach is exploited to adjust the weights of ANNs such that the approximated solution satisfies the FDE. Different types of FDEs including linear and nonlinear terms are solved to illustrate the ability of the method. In addition, the present scheme is compared with the analytical solution and a number of existing numerical techniques to show the efficiency of ANNs with high accuracy, fast convergence and low use of memory for solving the FDEs.

1. Introduction

In recent years, the fractional calculus has gained a great development in both theory and application. Its appearance and development, to a certain extent, make up for defects of the classical calculus of integer order. The fractional calculus has been used to describe many phenomenons in almost all applied sciences, such as fluid flow in porous materials, anomalous diffusion transport, acoustic wave propagation in viscoelastic materials, signal processing, nanotechnology, financial theory, electric conductance of biological systems and others (see, [5–9,44]).

Fractional differential equation (FDE) is a unifying theory expressing memory and hereditary characteristics of diverse materials and process in comparison with classical integer order derivative [8,24,30]. However, the progress of numerical techniques in this area has received huge attention and has undergone a rapid growth in recent years. These methods include spectral methods [3,4,10,11,12,14,15,22], fractional linear multi-step methods [19,43], Adomian decomposition method [31,32], Wavelet Method [23], Laplace transforms [40,41] and variational iteration method [35].

* Corresponding author at: Department of Mathematics, Cankaya University, 06530 Balgat, Ankara, Turkey.
E-mail addresses: pakdaman@mshdiau.ac.ir (M. Pakdaman), ahmadian.hosseini@gmail.com (A. Ahmadian), s-effati@um.ac.ir (S. Effati), soheilsalahshour@yahoo.com (S. Salahshour), dumitru@cankaya.edu.tr, baleanudumitru@yahoo.com (D. Baleanu).

http://dx.doi.org/10.1016/j.amc.2016.07.021
0096-3003/© 2016 Elsevier Inc. All rights reserved.

Establishing the aforementioned demands, we aims to illustrate the solution behavior of the artificial neural networks (ANNs) approximations for multi-term fractional differential equations (FDEs) of the form

$$\left({}_a^C D_t^{\alpha} x\right)(t) = f(t, x(t)), \quad a \le t \le b, \tag{1.1}$$

provided that the initial conditions

$$y^{(i)}(0) = \gamma_i, \quad i = 0, \ldots, n-1, \tag{1.2}$$

where $m-1 < \alpha \le m,\ m \in \mathbb{N} := \{1, 2, \ldots\},\ \gamma_i \in \mathbb{R} := (-\infty, \infty),\ {}_a^C D_t^{\alpha}$ is the Caputo-type derivative and $f : [0, b] \times \mathbb{R} \to \mathbb{R}$ is a given continuous function. To interpret the performance and behavior of a fractional dynamic systems, it is essential to exploit a suitable definition of the derivative of fractional order. We have several types of fractional derivatives such as Riemann–Liouville, Caputo, Grünwald–Letnikov, Riesz and Weyl that all these fractional derivative order definitions have their advantages and disadvantages. Among these fractional derivatives, the Riemann–Liouville and Caputo derivatives are the most common and popular fractional derivatives. However, the Riemann–Liouville derivative has certain disadvantages when trying to model real-world phenomena with FDEs. The Riemann–Liouville derivative of a constant is not zero. In addition, if an arbitrary function is a constant at the origin, its fractional derivation has a singularity at the origin for instant exponential and Mittag–Leffler functions. Theses disadvantages reduce the field of application of the Riemann–Liouville fractional derivative. Besides that, the main advantage of the Caputo's approach, which makes it more popular is that the initial conditions for the FDEs with the Caputo takes on the same form as for integer-order differential equations [20,21,36,48]. Although the structure of the approximate solution based on the present method is such that there is no requirement to have constraints for initial conditions satisfaction, in order to design the artificial neural network (ANN), it is more reasonable to use such type of differentiability.

In recent years, ANNs were attained a considerable attention as robust and effective tools for function approximation [1,2,25,33]. It is proved that a multi-layer perceptron can approximate continuous functions over a compact subset of R^n. Cybenko [13] presented a theorem to demonstrate the capabilities of the ANNs-based function approximation for the sigmoid activation functions. In a wide range of mathematical problems, we look for an unknown continuous function, satisfying some conditions. For example, in ordinary and partial differential equation theory as well as in the theory of integral equations and optimal control theory. For that reason, the work of Lagaris et. al [27] has an important rule in literature for approximating the solution of ordinary and partial differential equations. The application of ANNs for solving differential equations was not limited to ordinary and partial differential equations. Effati and Pakdaman [16] used the ability of ANNs-based function approximation, for solving fuzzy differential equations. They also in Effati and Pakdaman [17] used the ANNs methodology to approximate the state, co-state and control functions for an optimal control problem. Effati and Buzhabadi [18] used feed-forward ANNs to approximate the solution of Fredholm integral equations of the second kind. Using a ANN methodology can offer a differentiable continuous solution of differential equation, which satisfies all initial or boundary conditions, while the accuracy of the method can be adjusted by changing the parameters of ANN. Hence, Sabouri et al. [39] employed the ability of neural networks for solving fractional order optimal control problems. Jafariana et al. [26] presented an ANN approach for solving a class of FDEs. They used an unsupervised back-propagation learning algorithm for adjusting the weights of ANN and a suitable truncated power series of the solution function. In the current approach, we solve the FDEs directly using an unconstrained optimization problem which can be solved by any optimization technique. The trial solutions of the proposed ANN involve a single independent variable regardless of the dimension of the problem. The approximate solutions are continuous over all the domain of integration.

Recently, Raja et al. [45,46] presented a stochastic computational intelligence approach based on the strength of feed forward ANNs and Genetic algorithm for the solution of FDEs. However, for nonlinear FDEs, the accuracy is low. Thereafter, Raja et al. [47] proposed ANNs and sequential quadratic programming for solving fractional order Riccati equations. However, performance of perceptron ANNs, motivated the authors to use them to approximate the solution of FDEs. One of important advantages of the present ANNs for the function approximation is that they propose the solution of differential equations as a differentiable function. Although the ANN trains at a number of points, the solution can be calculated at each arbitrary point in training interval even between training points. To the best of authors' knowledge, such approach for solving the FDEs has not been reported so far in the literature which partially motivated us to propose this efficient and applicable technique for the solution of multi-terms FDEs.

To summarize, solving FDEs with the corresponding trained ANN technique offers the following preferences in comparison with the classical numerical schemes:

1. Solution search proceeds without coordinate transformations.
2. ANN learns to solve the DE analytically.
3. Computational complexity does not increase quickly when the number of sampling points increase.
4. Rapid calculation of the solution values.

Contrary to the above advantages, standard numerical methods such as predictor-correction method [49], Runge–Kutta method [51], fractional Euler method [34] and operational matrix methods [22,38] require the discretization of domain into the number of finite domains/points where the functions are approximated locally and their computational complexity increases rapidly with the number of sampling points [37,42]. Besides that, rounding-off errors solemnly influence the solution precision in the numerical techniques with complicatedness that also raise quickly with the number of sampling points [37].

Moreover, such numerical methods are usually iterative in nature, where we fix the step size before initiating the computation. After the solution is obtained, if we want to know the solution in between steps then again the procedure is to be repeated from initial stage. ANN may be one of the reliefs where we may overcome this repetition of iterations [50].

The rest of this paper is structured as follows. We briefly describes the fractional calculus definitions required in this study in Section 2. Afterwards, Section 3 presents the proposed ANN with the complete details. To validate the present method, we have considered various numerical examples including nonlinear and linear FDEs in Section 4. In this section, the method is compared with a number of numerical methods to demonstrate the efficiency of the present approach. Also, the error analysis and the convergence of the present technique for each of examples is presented based on the different values of fractional derivative. Finally, Section 5 draws some conclusions from the results and presents a number of ideas for future works.

2. Preliminaries and notations

In this section, we present some notations, definitions and preliminary facts of the fractional calculus theory which can be found in [7,36,48].

Definition 2.1. Let $[a, b]$ be a finite interval on the real axis $\mathbb{R}$. The Riemann–Liouville integral $({}_aI_t^\alpha x)(t)$ and the Riemann–Liouville fractional derivative $({}_aD_t^\alpha x)(t)$ of order $\alpha > 0$ are defined by

$$({}_aI_t^\alpha x)(t) = \frac{1}{\Gamma(\alpha)}\int_a^t (t-\tau)^{\alpha-1}x(\tau)\,d\tau, \quad t > a,$$

and

$$({}_aD_t^\alpha x)(t) = \frac{1}{\Gamma(m-\alpha)}\frac{d^m}{dt^m}\int_a^t (t-\tau)^{m-\alpha-1}x(\tau)\,d\tau, \quad t > a,$$

respectively, where $m-1 < \alpha \le m, m \in \mathbb{N}$, and $\Gamma(.)$ denotes the Gamma function.

As it was stated in the introduction section, its derivative has certain disadvantages when trying to model real-world phenomena with FDEs. In fact, Caputo derivative implies a memory effect by means of a convolution between the integer order derivative and a power of time [38]. Therefore, in this research we prefer to use the Caputo fractional derivative.

Definition 2.2. Let $[a, b]$ be a finite interval on the real axis $\mathbb{R}$. The Caputo fractional derivative $({}_a^CD_t^\alpha x)(t)$ of order α is defined by

$$({}_a^CD_t^\alpha x)(t) = {}_aD_t^\alpha\left(x(t) - \sum_{i=0}^{m-1}\frac{x^{(i)}(a)}{i!}(t-a)^i\right), \quad t > a,\ m-1 < \alpha \le m,\ m \in \mathbb{N}, \tag{2.3}$$

where ${}_aD_t^\alpha$ is the Riemann–Liouville fractional derivative. Note that if $x^{(i)}(a) = 0, i = 0, 1, \ldots, m-1$, then $({}_a^CD_t^\alpha x)(t)$ coincides with $({}_aD_t^\alpha x)(t)$. Further, if x(t) is continuously differentiable on $[a, b]$ up to order m, then the expression (2.3) can be reduced to

$$({}_a^CD_t^\alpha x)(t) = \frac{1}{\Gamma(m-\alpha)}\int_a^t (t-\tau)^{m-\alpha-1}x^m(\tau)\,d\tau, \quad t > a,\ m-1 < \alpha \le m,\ m \in \mathbb{N},$$

which is sometimes called a smooth fractional derivative.

The Caputo fractional differentiation is a linear operation

$${}_a^CD_t^\alpha(\lambda f(x) + \mu g(x)) = \lambda({}_a^CD_t^\alpha f)(x) + \mu({}_a^CD_t^\alpha g)(x),$$

where λ and μ are constants. Also, we have

$${}_a^CD_t^\alpha C = 0, \quad (C \text{ is a constant}),$$

$${}_a^CD_t^\alpha x^\beta = \begin{cases} 0, & \textit{for } \beta \in \mathbb{N}_0 \textit{ and } \beta < \lceil\alpha\rceil, \\ \frac{\Gamma(\beta+1)}{\Gamma(\beta+1-\alpha)}x^{\beta-\alpha}, & \textit{for } \beta \in \mathbb{N}_0 \textit{ and } \beta \ge \lceil\alpha\rceil \textit{ or } \beta \notin \mathbb{N} \textit{ and } \beta > \lfloor\alpha\rfloor. \end{cases}$$

The ceiling function $\lceil\alpha\rceil$ is used to denote the smallest integer greater than or equal to α, and the floor function $\lfloor\alpha\rfloor$ to denote the largest integer less than or equal to α. Also $\mathbb{N} = \{1, 2, \ldots\}$ and $\mathbb{N}_0 = \{0, 1, 2, \ldots\}$. Moreover, it has the following two basic properties for $m-1 < \alpha \le m$ and $x(t)$ belongs to the Lebesgue space $L_1[a, b]$,

$$({}_a^CD_t^\alpha\, {}_aI_t^\alpha x)(t) = x(t),$$

and

$$({}_aI_t^\alpha\, {}_a^CD_t^\alpha x)(t) = f(t) - \sum_{k=0}^{m-1}x^{(k)}(0^+)\frac{(t-a)^k}{k!}, \quad t > 0.$$

3. Solution method

In the theory of perceptron ANNs, it is proved that they can approximate continuous functions. In this section, at first, we recall an important theorem which deals with the ability of ANNs for the function approximation, afterwards, our proposed ANN technique is presented for solving FDEs.

Definition 3.1 (See, [13]). We say that the single variable function $f(t)$ is sigmoidal if:

$$f(t) \to \begin{cases} 1 & \text{as } t \to +\infty \\ 0 & \text{as } t \to -\infty \end{cases} . \tag{3.4}$$

Theorem 3.1 (See, [13]). *Let f be any continuous sigmoidal function. Then finite sums of the form:*

$$G(x) = \sum_{j=1}^{N} \alpha_j f(y_j^T x + b_j), \tag{3.5}$$

are dense in the space of continuous functions defined over an interval I (denoted by C(I)). In other words, considering any continuous function f and $\epsilon > 0$, *there is a sum* $G(x)$, *of the above form which*

$$|G(x) - f(x)| < \epsilon, \quad \textit{for all } x \in I. \tag{3.6}$$

Now, let us consider the following FDE:

$$({}_{t_0}^{C}D_t^{\alpha} x)(t) = f(t, x(t)), x(t_0) = x_0, \tag{3.7}$$

where $t_0 \leq t \leq T$. Suppose that $x_N(t, \mathbf{\Omega})$ is an approximate solution of the problem (3.7). The structure of x_N has two important specialities: first, it contains an ANN with $\mathbf{\Omega}$ as a vector containing all it's corresponding weights and second is that it must satisfies the initial condition of problem (3.7). In the other word, if $\mathbf{\Omega}^*$ contains the optimal values of the weights of the ANN, then we must have: $x_N(t_0, \mathbf{\Omega}^*) = x_0$. Mathematical formulation of $x_N(t, \mathbf{\Omega})$ is as follows:

$$x_N(t, \mathbf{\Omega}) = x_0 + (t - t_0)N(t, \mathbf{\Omega}), \tag{3.8}$$

where

$$N(t, \mathbf{\Omega}) = \sum_{k=1}^{n} \nu_k \varphi(\zeta_k), \tag{3.9}$$

in which φ is activation function and $\zeta_k = w_k t + \beta_k$ and n is the number of nodes in hidden layer. In this paper, we use the sigmoid transfer function $\varphi(x) = \frac{1}{1+e^{-x}}$. It is obvious that the structure of the relation (3.9) is similar to the Eq. (3.5). Thus, using Theorem 3.1 assures us that the Eq. (3.8) can approximate the solution of the FDE (3.7), properly. Substituting the Eq. (3.8) in the Eq. (3.7), we have:

$$({}_{t_0}^{C}D_t^{\alpha} x_N)(t, \mathbf{\Omega}) = f(x_N(t, \mathbf{\Omega}), t). \tag{3.10}$$

It is obvious that $x_N(t, \mathbf{\Omega})$ satisfies the initial condition. Instead of solving the Eq. (3.7), we solve the Eq. (3.10). So, at first, we discretize the interval $[t_0, T]$ to m subintervals, then, solve the following optimization problem:

$$\min_{\mathbf{\Omega}} R(\mathbf{\Omega}) = \sum_{i=1}^{m} [({}_{t_0}^{C}D_t^{\alpha} x_N)(t_i, \mathbf{\Omega}) - f(x_N(t_i, \mathbf{\Omega}), \tau)]^2. \tag{3.11}$$

To approximate the fractional derivative in the Eq. (3.10), as it was stated in Section 2, we use the Caputo derivative such that:

$$({}_{t_0}^{C}D_t^{\alpha} x_N)(t_i, \mathbf{\Omega}) = \frac{1}{\Gamma(m-\alpha)} \int_0^{t_i} (t_i - \tau)^{m-\alpha-1} x_N^{(m)}(\tau, \mathbf{\Omega}) d\tau, \tag{3.12}$$

where the integral can be approximated by any numerical integration method. Here, we employ the trapezoidal integration rule. For constant values of α, m and for sake of simplicity, we denote:

$$I_i(t_i, \mathbf{\Omega}) = \frac{1}{\Gamma(m-\alpha)} \int_0^{t_i} (t_i - \tau)^{m-\alpha-1} x_N^{(m)}(\tau, \mathbf{\Omega}) d\tau, \tag{3.13}$$

also:

$$J(t_i, \tau, \mathbf{\Omega}) = (t_i - \tau)^{m-\alpha-1} x_N^{(m)}(t_i, \mathbf{\Omega}). \tag{3.14}$$

Thus, we have:

$$I_i(t_i, \mathbf{\Omega}) = \frac{1}{\Gamma(m-\alpha)} \int_0^{t_i} J(t_i, \tau, \mathbf{\Omega}) d\tau. \tag{3.15}$$

By considering the trapezoidal rule and discretizing the interval [0, t_i] into the k sub-interval ($h_i = t_i/k$), we have:

$$I_i(t_i, \mathbf{\Omega}) \simeq \frac{h_i}{2\Gamma(m-\alpha)}[J(0,\tau,\mathbf{\Omega}) + 2J(h_i,\tau,\mathbf{\Omega}) + \cdots + 2J((k-1)h_i,\tau,\mathbf{\Omega}) + J(t_i,\tau,\mathbf{\Omega})]. \tag{3.16}$$

The approximate value of $I_i(t_i, \mathbf{\Omega})$, obtained by the trapezoidal rule, is indicated by $\tilde{I}(t_i, \mathbf{\Omega})$ in the remainder of the paper. Thus, from the Eq. (3.12) we have:

$$({}^{C}_{t_0}D^{\alpha}_{t}x_N)(t_i, \mathbf{\Omega}) = \tilde{I}(t_i, \mathbf{\Omega}). \tag{3.17}$$

Now by substituting the Eq. (3.17) into the Eq. (3.11), the following optimization problem is achieved:

$$\min_{\mathbf{\Omega}} R(\mathbf{\Omega}) = \sum_{i=1}^{m}[\tilde{I}(t_i, \mathbf{\Omega}) - f(x_N(t_i, \mathbf{\Omega}), t_i)]^2. \tag{3.18}$$

Note that the Eq. (3.18) is an unconstrained optimization problem. To solve the optimization problem, we can apply the classical optimization methods or heuristic algorithms. In this research, we exploit the Broyden–Fletcher–Goldfarb–Shanno (BFGS) Quasi–Newton method which is quadratically convergent. Suppose that $\mathbf{\Omega}^*$ is the optimal solution of the optimization problem (3.11). By replacing $\mathbf{\Omega}^*$ into the approximation solution (3.8), the numerical solution of the problem (3.7), $x_N(t, \mathbf{\Omega}^*)$, is achieved.

To summarize, the main features of the proposed ANN method are listed as follows:

- Based on the structure of the proposed ANN method (Eqs. (3.8) and (3.9)), the obtained solution of the method, for FDEs, is a continuous and differentiable function.
- The precision of the scheme can be grown up by increasing the number of neurons or using any optimization method such as heuristic algorithms.
- The value of the solution can be obtained at any arbitrary point even between training points. However, outside the training interval, the error increases. The solution of the FDE is available for each arbitrary point in the training interval (even between the training points). Indeed, solving the FDE results an approximate function, therefore, it is possible to find the approximate solution at every point.
- Unlike the most of the numerical methods, ANN can be applied for linear, non-linear and system of FDEs as well as for multi-derivative FDEs.

4. Numerical simulations

In this section, in order to demonstrate the effectiveness of the proposed ANN, we consider numerical examples of linear and nonlinear nature, for which, either closed form solution exist or numerical solutions by other methods are known. We use five neuron for the proposed ANN technique. In the optimization step, we apply the BFGS Quasi–Newton method [29] which is quadratically convergent. However, we can also use heuristic methods. The computation of solution for the following examples are performed in MATLAB 7 using the unconstrained optimization function *fminunc* via the BFGS hessian update.

It is worth mentioning that for all examples, a multi-layer perceptron consisting of one hidden layer with ten hidden units and one linear output unit is used. In order to obtain higher accuracy (especially in the nonlinear cases), more hidden layers or training points can be employed. To depict the convergence of the weights, which are computed based on the present method, they are plotted over a number of iterations.

Example 4.1. Consider the following well-known fractional order differential equation (see, [38]):

$$({}^{C}_{0}D^{\alpha}_{t}x)(t) = -x(t), \ \ x(0) = 1, x'(0) = 0, \ \ t \in [0, 1] \tag{4.19}$$

The second initial condition is for $\alpha > 1$ only. The exact solution of this problem is as follows:

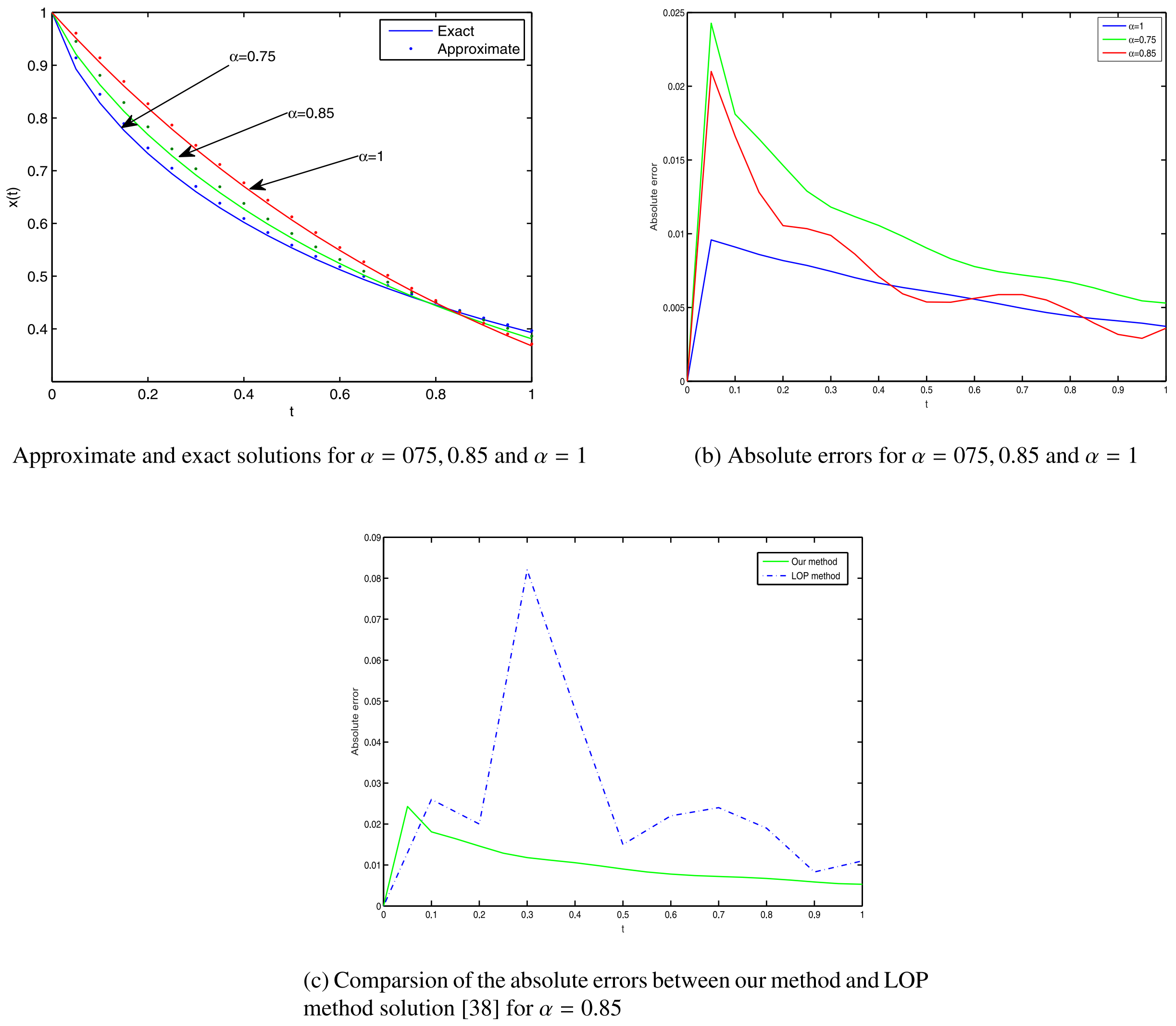
$$x(t) = \sum_{k=0}^{\infty} \frac{(-t^{\alpha})^k}{\Gamma(\alpha k + 1)}.$$

We propose the approximate solution as:

$$x_N(t, \mathbf{\Omega}) = 1 + t^2 N(t, \mathbf{\Omega}).$$

Fig. 1a and b show that the numerical results from our proposed method are in good agreement with the exact solutions for the different values of α. Moreover, a comparison between the accuracy of our method and the Legendre operational matrix (LOP) method [38] is illustrated in Fig. 1c for $\alpha = 0.85$ when $t \in [0, 1]$. From Fig. 1, it can be observed that the error rate is decreasing when α is approaching 1. The results verifies the reliability of the proposed method in the whole of the time interval.

To emphasize the convergence behavior of the computed values of the weight parameters of the layers, Fig. 2 is graphed for the input layer, bias and output weights for $\alpha = 0.85$.



(a) Approximate and exact solutions for α = 075, 0.85 and α = 1

(b) Absolute errors for α = 075, 0.85 and α = 1

(c) Comparsion of the absolute errors between our method and LOP method solution [38] for $\alpha = 0.85$

Fig. 1. Approximate solution is compared with the exact solution and LOP method [38] over $t = [0, 1]$ for Example 4.1.

Example 4.2. Consider the following nonlinear FDE (see, [34]):

$$({}_0^C D_t^\alpha x)(t) = x^2(t) - \frac{2}{(t+1)^2}, \; x(0) = -2, \; t \in [0, 1]. \tag{4.20}$$

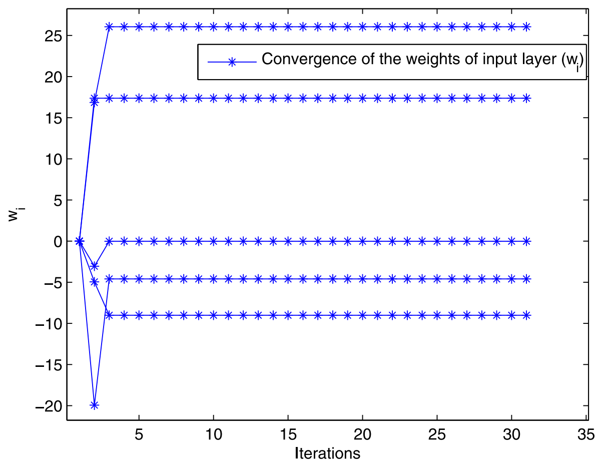
The analytical solution of the above problem for $\alpha = 1$ is $x(t) = \frac{-2}{t+1}$.

The approximate solution of the problem (4.20) based on the proposed ANN that holds the initial conditions is as follows:

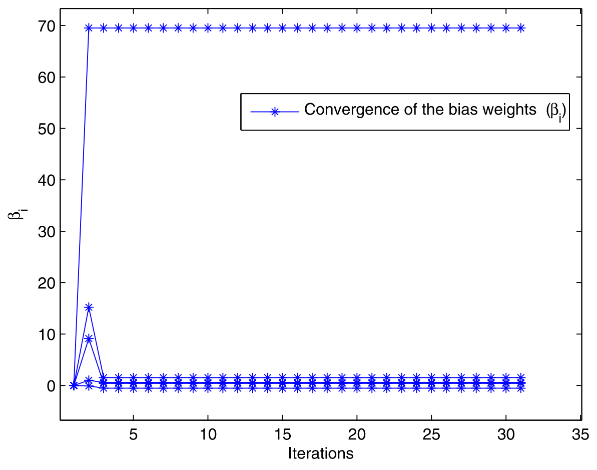
$$x_N(t, \mathbf{\Omega}) = -2 + tN(t, \mathbf{\Omega}).$$

The numerical solution is compared with the exact solution for $\alpha = 1$ in Fig. 3a. Note that the analytical and numerical solutions are roughly coincided. From Fig. 3c, we can see our numerical solution in comparison with the solutions achieved by predictor-corrector (PC) method [49] and fractional Euler (FC) method [34] for $\alpha = 0.6$. The comparison among their absolute errors is shown in Fig. 3c for $\alpha = 1$. To have more details of the results, the numerical outputs with comparison to Ref. [34] and exact solution is given in Table 1 for $\alpha = 0.4$, 0.8 and 1 on the interval [0, 1]. From Fig. 3 and Table 1, we infer that our method provides better accuracy than the given in Refs. [34,49]. Therefore, the present ANN is a valid method for solving FDEs.

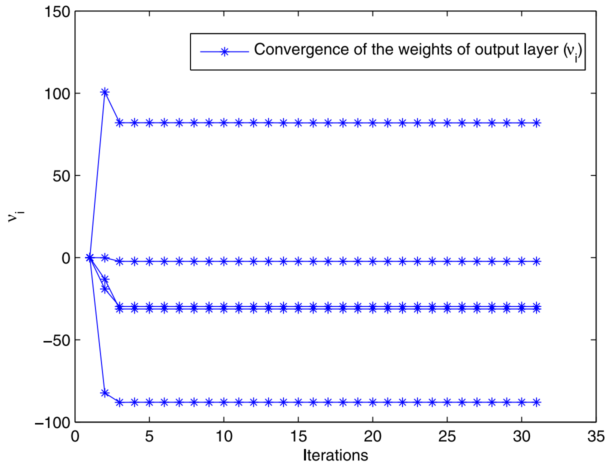
Furthermore, the convergence behaviors of the weights of input layer, bias and the weights of output layer (w_i, β_i and ν_i) are displayed in Fig. 4 for $\alpha = 0.4$.



(a) Convergence of the weights of input layer



(b) Convergence of bias weights



(c) Convergence of the out put weights

Fig. 2. Convergence of the input, output and bias weights for Example 4.1 for $\alpha = 0.85$.

Table 1
Numerical results of Example 4.2 with comparison to the solution of the FE method [34] over $t = [0, 1]$.

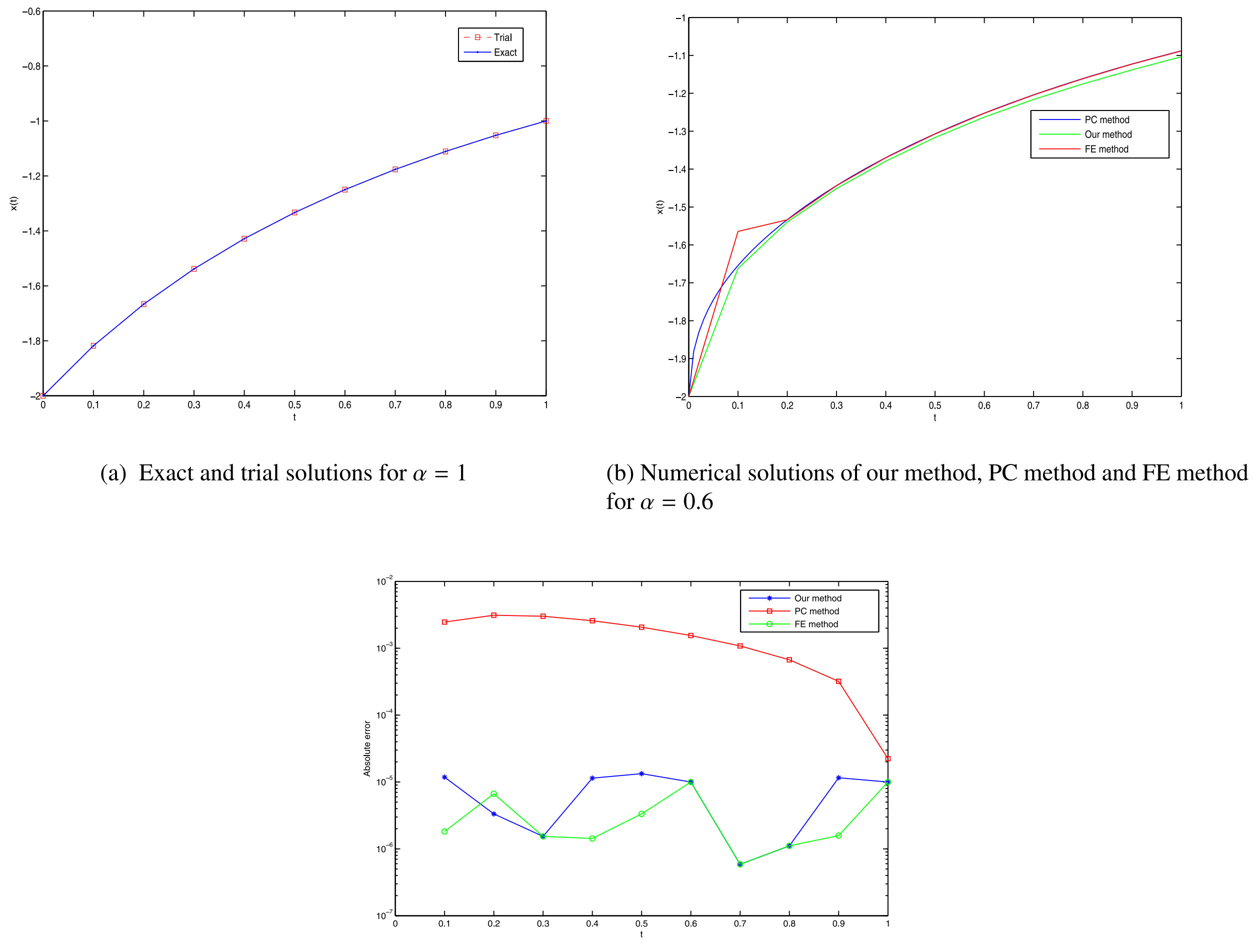
t	$\alpha = 0.4$		$\alpha = 0.8$		$\alpha = 1$		
	Our method	FE method	Our method	FE method	Our method	FE method	Exact
0	−2.00000	−2.00000	−2.00000	−2.00000	−2.00000	−2.00000	−2.00000
0.2	−1.49910	−1.48991	−1.59475	−1.59189	−1.66667	−1.66666	−1.66667
0.4	−1.36851	−1.35254	−1.39611	−1.39284	−1.42857	−1.42857	−1.42857
0.6	−1.27892	−1.25413	−1.25308	−1.24987	−1.25000	−1.25000	−1.25000
0.8	−1.21037	−1.17879	−1.14291	−1.13896	−1.11111	−1.11111	−1.11111
1	−1.15920	−1.11893	−1.05394	−1.04957	−0.99999	−0.99999	−1.00000

Example 4.3. Consider the following FDE (see, [28]):

$$({}_0^C D_t^{\alpha} x)(t) = -x(t) + t^2 + \frac{2t^{2-\alpha}}{\Gamma(3-\alpha)}, \; x(0) = 0, \; t \in [0, 1]. \tag{4.21}$$

The accuracy solution in this case is given by

$$x(t) = t^2.$$



(a) Exact and trial solutions for $\alpha = 1$

(b) Numerical solutions of our method, PC method and FE method for $\alpha = 0.6$

(c) Absolute error of our solution in comparsion with the solutions of the PC method and the FE method for $\alpha = 1$

Fig. 3. Approximate solution of our method in comparison with the exact solution, and the solutions of the PC method [10] and FE method [11] over $t = [0, 1]$ for Example 4.2.

The proposed ANN-based approximate solution of the Eq. (4.21) that satisfy the initial conditions is as follows:

$$x_N(t, \mathbf{\Omega}) = tN(t, \mathbf{\Omega}).$$

The approximate solutions for this problem are obtained by using the PC method, reported in [49], and our proposed ANN scheme. Their absolute errors are given in Fig. 5a and b, respectively. It is obvious that our method provides much superior result than the PC method for $\alpha = 0.75, 0.85$ and 0.95 while for $\alpha = 1$ both methods obtained same order of the accuracy (however we can employ more training points or more weights to achieve more excellent outputs).

Once again, the convergence properties of the weights of input layer, bias and the weights of output layer (w_i, β_i and ν_i) are plotted in Fig. 6 for $\alpha = 0.75$.

Example 4.4. We consider the following initial value problem in the case of the inhomogeneous Bagley–Torvik equation (see, [46]):

$$D^{(2)}x(t) + ({}_0^C D_t^{(1.5)}x)(t) + x(t) = 1 + t, \;\; x(0) = 1, x'(0) = 1, \;\; t \in [0, 1]. \tag{4.22}$$

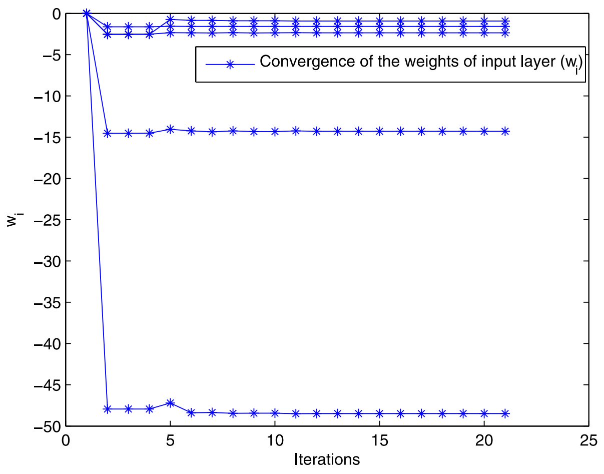
The exact solution of this problem is $x(t) = 1 + t$.

The proposed approximate solution based on the ANN technique is as:

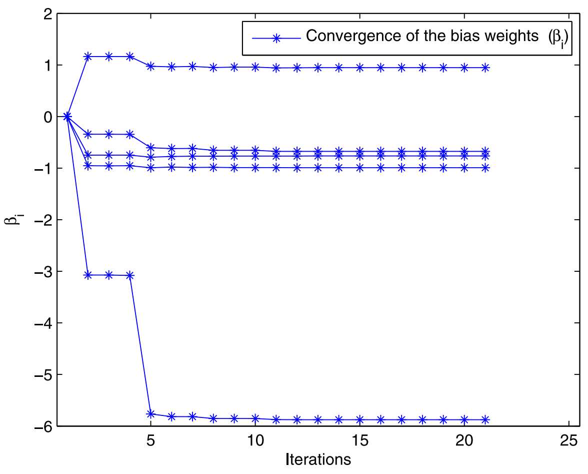
$$x_N(t, \mathbf{\Omega}) = 1 + t + t^2 N(t, \mathbf{\Omega}).$$

It is easy to examine that the approximate solution satisfies the initial conditions.

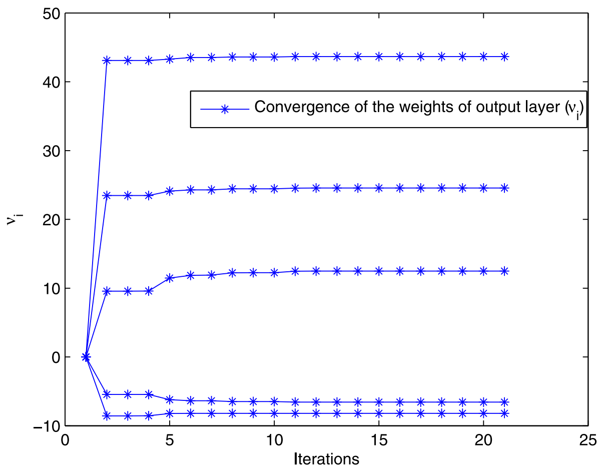
By substituting the approximate solution $x_N(t, \mathbf{\Omega})$ into the Eq. (4.22) and using the same parameters as provided in Example 4.3, (training the network for 30 equidistant points in the domain [0, 1] and also using $n = 5$ neurons), the



(a) Convergence of the weights of input layer



(b) Convergence of bias weights



(c) Convergence of the out put weights

Fig. 4. Convergence of the input, output and bias weights for Example 4.2 for $\alpha = 0.4$.

following optimization problem is acquired:

$$\min_{\mathbf{\Omega}} R(\mathbf{\Omega}) = \sum_{i=1}^{20} [D^{(2)} x_N(t_i, \mathbf{\Omega}) + ({}^{C}_{t_0}D_t^{(1.5)} x_N)(t_i, \mathbf{\Omega}) + x_N(t_i, \mathbf{\Omega}) - 1 - x_N(t_i, \mathbf{\Omega})]^2, \tag{4.23}$$

where

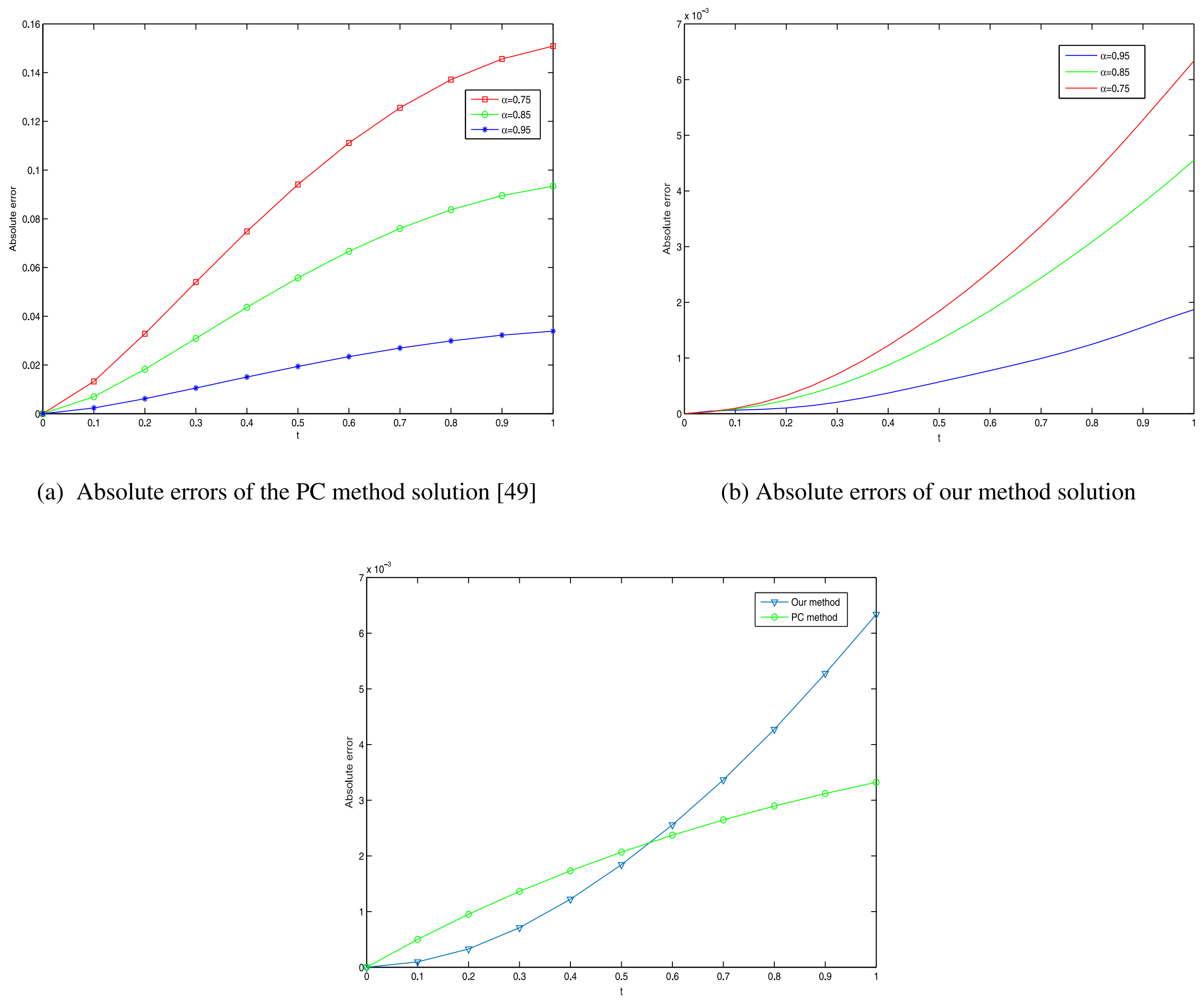
$$x_N(t, \mathbf{\Omega}) = 1 + t + t^2 N(t, \mathbf{\Omega}),$$

and

$$D^{(2)} x_N(t_i, \mathbf{\Omega}) = 2N(t_i, \mathbf{\Omega}) + 4t_i \frac{\partial N}{\partial t}(t_i, \mathbf{\Omega}) + t_i^2 \frac{\partial^2 N}{\partial t^2}(t_i, \mathbf{\Omega}).$$

Now, to approximate $({}^{C}_{t_0}D_t^{(1.5)} x_N)(t_i, \mathbf{\Omega})$, we use the Caputo formula (3.12) and finally to solve expression (4.23), the BFGS Quasi–Newton method is applied.

In Table 2 we introduce the approximate solutions for Example 4.4 using our method and a stochastic technique [46] with the help of feed-forward ANN based on the genetic algorithm (GA) hybrid and pattern search (PS) technique [46,47] over $t \in [0, 1]$. we imply that our method provides much superior result than given in [46]. Also from Fig. 7 again, it is confirmed that the absolute error function has a smoother behavior and higher accuracy than the stochastic technique throughout the reported interval. This demonstrates the importance of our scheme in solving Bagley–Torvik equations.



(a) Absolute errors of the PC method solution [49]

(b) Absolute errors of our method solution

(c) Absolute errors of our method solution compared with PC method solution [49] for $\alpha = 1$

Fig. 5. Absolute errors of the solution of our method in comparison with the PC method solution [49] using different values of α over $t = [0, 1]$ for Example 4.3.

Besides that, the convergence behaviors of the weights of input layer, bias and the weights of output layer (w_i, β_i and ν_i) are plotted in Fig. 8a–c, respectively.

Example 4.5. Consider the following nonlinear initial value problem (see, [22]):

$$D^{(3)}x(t) + ({}_0^C D_t^{(2.5)}x)(t) + x^2(t) = t^4,\ \ x(0) = x'(0) = 0, x''(0) = 2\ t \in [0, 1]. \tag{4.24}$$

The exact solution is $x(t) = t^2$. To satisfy the initial conditions, we choose the following approximate solution

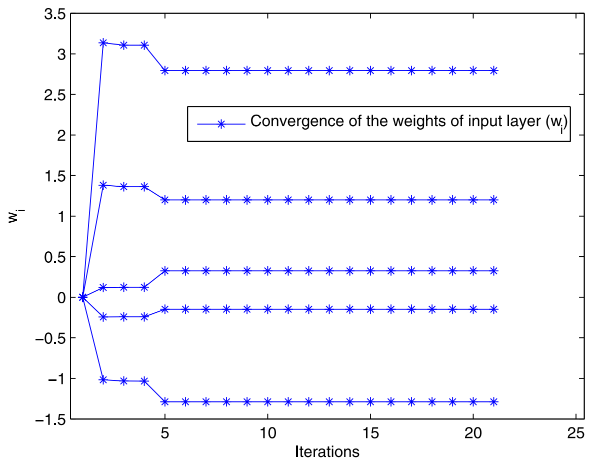
$$x_N(t, \mathbf{\Omega}) = t^2 + t^3 N(t, \mathbf{\Omega}).$$

By replacing the above approximate solution into the Eq. (4.24), training the interval [0, 1] to $m = 20$ points and also using $n = 5$ neurons, we get the following unconstrained optimization problem:

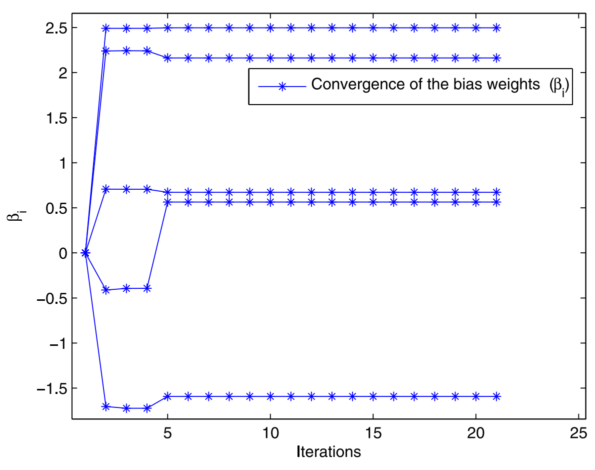
$$\min_{\mathbf{\Omega}} R(\mathbf{\Omega}) = \sum_{i=1}^{20} [D^{(3)}x_N(t_i, \mathbf{\Omega}) + ({}_{t_0}^C D_t^{(2.5)}x_N)(t_i, \mathbf{\Omega}) + x_N^2(t_i, \mathbf{\Omega}) - x_N^4(t_i, \mathbf{\Omega})]^2, \tag{4.25}$$

where

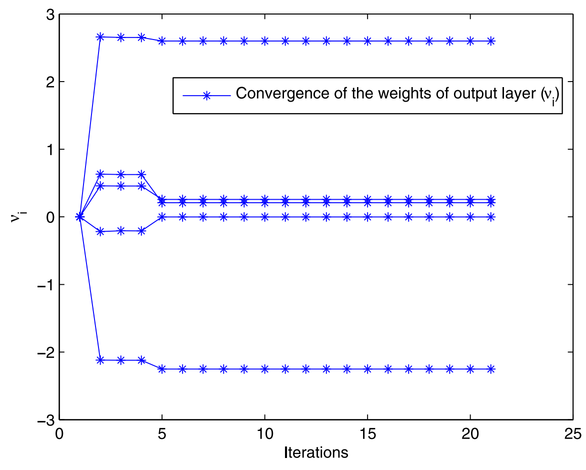
$$x_N(t, \mathbf{\Omega}) = t^2 + t^3 N(t, \mathbf{\Omega})$$



(a) Convergence of the weights of input layer



(b) Convergence of bias weights



(c) Convergence of the out put weights

Fig. 6. Convergence of the input, output and bias weights for Example 4.3 for $\alpha = 0.75$.

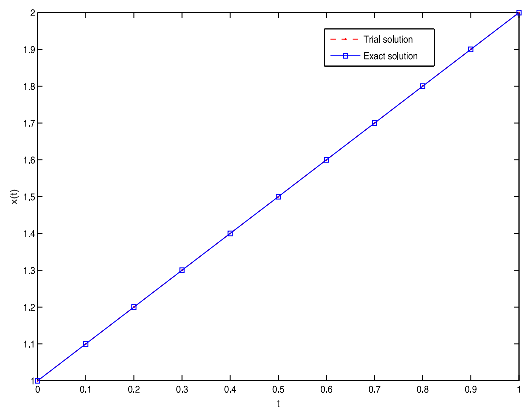
Table 2
Comparison of the absolute errors of our method, E_{ANN}, with the stochastic technique (GA-based), E_{GA}, and (GA-PS-based), E_{GP}, [46] over $t = [0, 1]$ for Example 4.4.

t	Exact	Our method	GA-based [46]	GA-PS-based [46]	E_{ours}	E_{GA}[46]	E_{GP}[46]
0	1.00	1.000000	1.024862	1.016007	0	2.30e−2	1.60e−2
0.2	1.20	1.199997	1.220821	1.199804	2.18e−6	3.13e−2	1.95e−4
0.4	1.40	1.399996	1.426952	1.401629	3.24e−6	3.45e−2	1.62e−3
0.6	1.60	1.599997	1.634569	1.607429	2.91e−6	1.36e−2	7.42e−3
0.8	1.80	1.799997	1.828738	1.799987	2.71e−6	2.30e−2	1.27e−5
1	2.00	1.999995	1.985057	1.953762	4.03e−6	3.13e−2	4.62e−2

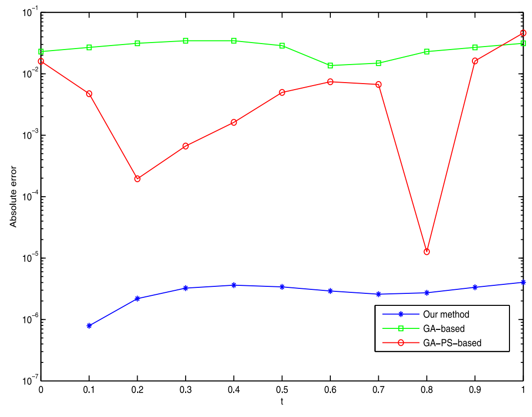
and

$$D^{(3)}x_N(t_i, \mathbf{\Omega}) = \frac{\partial^3 x_N}{\partial t^3}(t_i, \mathbf{\Omega}).$$

To approximate $({}^{C}_{t_0}D^{(2.5)}_t x_N)(t_i, \mathbf{\Omega})$, we use the formula (3.12) described in Section 3. The Eq. (4.25) is an unconstrained optimization problem, but the structure of the approximate solution is such that there is no requirement to have constraints for initial conditions satisfaction. To solve Eq. (4.25), we employ the BFGS Quasi–Newton algorithm, however we can use any optimization technique.

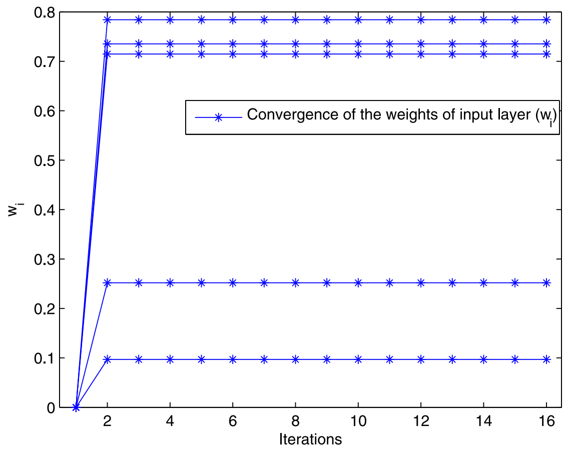


(a) Exact and trial solutions

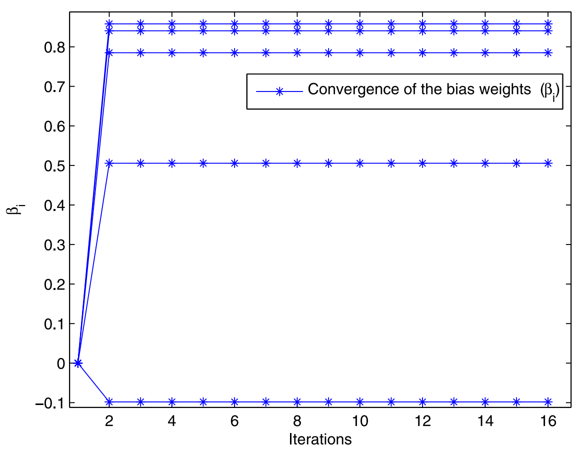


(b) Absolute errors of the solution of our method compared with the other methods

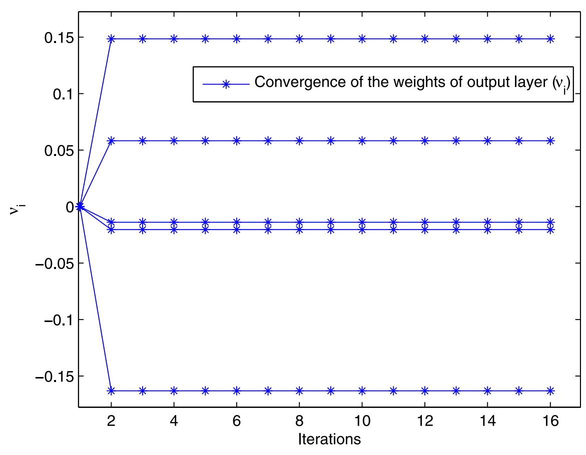
Fig. 7. Approximate solution of our method in comparison with the exact solution, and the solutions of the stochastic technique based on the GA and GA-PS [46] over $t = [0, 1]$ for Example 4.4.



(a) Convergence of the weights of input layer

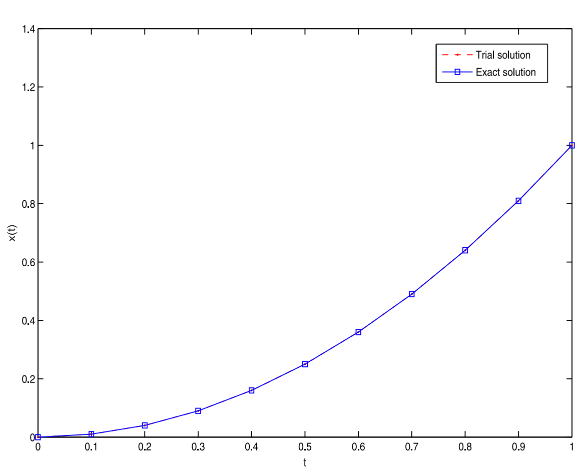


(b) Convergence of bias weights

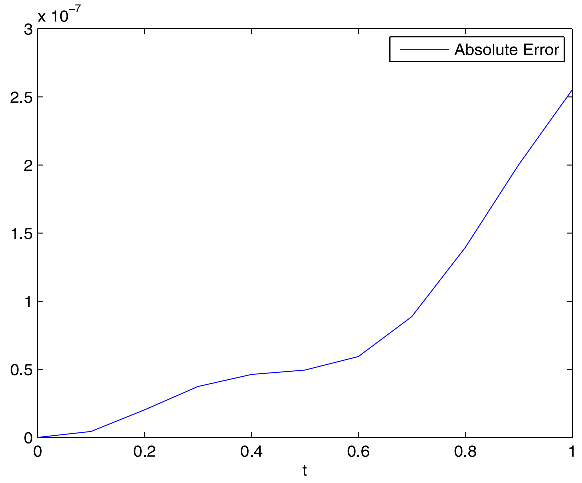


(c) Convergence of the out put weights

Fig. 8. Convergence of the input, output and bias weights for Example 4.4.

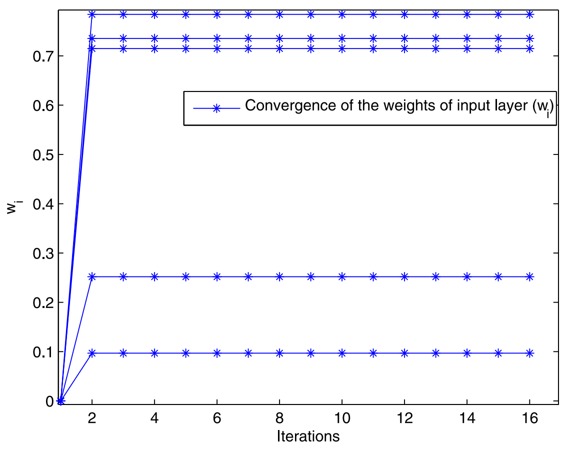


(a) Exact and trial solutions

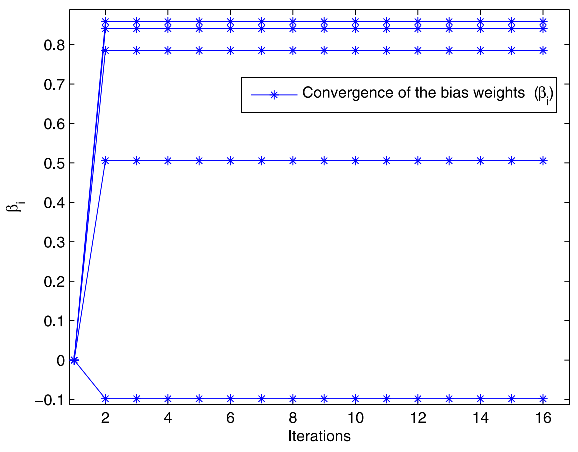


(b) Absolute errors of the solution of our method

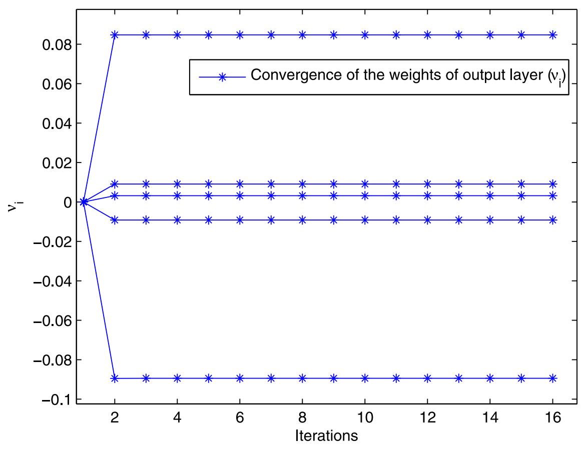
Fig. 9. Approximate solution of our method in comparison with the exact solution over $t = [0, 1]$ for Example 4.5.



(a) Convergence of the weights of input layer



(b) Convergence of bias weights



(c) Convergence of the out put weights

Fig. 10. Convergence of the input, output and bias weights for Example 4.5.

Comparisons between analytical and approximate results are depicted in Fig. 9a. Plot of the absolute error function is cited in Fig. 9b. Excellent agreement of the results between the analytical and the ANN-based approximate solution shows the powerfulness and reliability of the proposed method.

Also, the convergence behaviors of the weights of input layer, bias and the weights of output layer (w_i, β_i and ν_i) are illustrated in Fig. 10.

5. Conclusion

We developed a novel approach for solving FDEs which is formulated based on an ANN scheme. The effectiveness and applicability of this approach was validated by solving different types of multi-term FDEs. The outputs establish the good precision and rapid convergence of the proposed technique, which takes conveniences of the characteristics of the ANNs. This is as a result of the efficiency of the ANN method to approximate unknown functions and relations and also the trial solution which is in a close and differentiable form and satisfy into the initial conditions. We demonstrated, for the first time in the literature, the capability of the perceptron ANN to achieve the approximate solutions of FDEs which can reflect the behavior of dynamic models precisely in the engineering problems.

For our future research, we will develop the proposed approach for solving different types of FDEs such as fractional pantograph equations, fractional delay equations, fractional functional differential equations and etc. based on the Caputo–Fabrizio fractional derivative.

References

[1] S. Abbasbandy, M. Otadi, M. Mosleh, Numerical solution of a system of fuzzy polynomials by fuzzy neural network, Inf. Sci. 178 (2008) 1948–1960.
[2] S. Abbasbandy, M. Otadi, Numerical solution of fuzzy polynomials by fuzzy neural network, Appl. Math. Comput. 181 (2006) 1084–1089.
[3] A. Ahmadian, S. Salahshour, C.S. Chan, Fractional differential systems: a fuzzy solution based on operational matrix of shifted Chebyshev polynomials and its applications. IEEE Trans. Fuzzy Syst. Vol. PP, no. 99, pp. 1–1 doi:10.1109/TFUZZ.2016.2554156.
[4] A. Ahmadian, S. Salahshour, D. Baleanu, H. Amirkhani, R. Yunus, Tau method for the numerical solution of a fuzzy fractional kinetic model and its application to the oil palm frond as a promising source of xylose, J. Comput. Phys. 294 (2015) 562–584.
[5] R.L. Bagley, P.J. Torvik, On the appearance of the fractional derivative in the behavior of real materials, J. Appl. Mech. 51 (1994) 294–298.
[6] R.L. Bagley, P.J. Torvik, A theoretical basis for the application of fractional calculus to viscoelasticity, J. Rheol. 27 (1983) 201–210.
[7] D. Baleanu, K. Diethelm, E. Scalas, J.J. Trujillo, Fractional calculus models and numerical methods, Series on Complexity, Nonlinearity and Chaos, World Scientific, 2012.
[8] D. Baleanu, Z.B. Güvenc, J.A.T. Machado, New Trends in Nanotechnology and Fractional Calculus Applications, Springer-Verlag, 2010.
[9] E. Bazhlekova, I. Bazhlekov, Viscoelastic flows with fractional derivative model: computational approach by convolutional calculus of Dimovski, Fract. Calc. Appl. Anal. 17 (2014) 954–976.
[10] A.H. Bhrawy, M.M. Tharwat, A. Yildirim, A new formula for fractional integrals of Chebyshev polynomials: application for solving multi-term fractional differential equations, Appl. Math. Model. 37 (2013) 4245–4252.
[11] A.H. Bhrawy, D. Baleanu, L.M. Assas, Efficient generalized laguerre-spectral methods for solving multi-term fractional differential equations on the half line, J. Vib. Control 20 (2014a) 973–985.
[12] A.H. Bhrawy, Y.A. Alhamed, D. Baleanu, A.A. Al-Zahrani, New spectral techniques for systems of fractional differential equations using fractional-order generalized laguerre orthogonal functions, Fract. Calc. Appl. Anal. 17 (2014b) 1137–1157.
[13] G. Cybenko, Approximations by superpositions of sigmoidal functions, mathematics of control, Signals Syst. 2 (1989) 303–314.
[14] Y. Chen, Y. Sun, L. Liu, Numerical solution of fractional partial differential equations with variable coefficients using generalized fractional-order Legendre functions, Appl. Math. Comput. 244 (2014) 847–858.
[15] E.H. Doha, A.H. Bhrawy, S.S. Ezz-Eldien, A Chebyshev spectral method based on operational matrix for initial and boundary value problems of fractional order, Comput. Math. Appl. 62 (2011) 2364–2373.
[16] S. Effati, M. Pakdaman, Artificial neural network approach for solving fuzzy differential equations, Inf. Sci. 180 (2010) 1434–1457.
[17] S. Effati, M. Pakdaman, Optimal control problem via neural networks, Neural Comput. Appl. 23 (2012) 2093–2100.
[18] S. Effati, R. Buzhabadi, A neural network approach for solving Fredholm integral equations of the second kind, Neural Comput. Appl. 21 (2012) 843–852.
[19] N.J. Ford, A.C. Simpson, The numerical solution of fractional differential equations: speed versus accuracy, Numer. Algorithms 26 (2001) 333–346.
[20] J.F. Gómez-Aguilar, J.R. Razo-Hernández, J. Rosales-García, M. Guía-Calderón, Fractional RC and LC electrical circuits, Ingeniera Investigación y Tecnologa XV (2014) 311–319.
[21] J.F. Gómez-Aguilar, J.J. Rosales-García, J.J. Bernal-Alvarado, T. Córdova-Fraga, R. Guzmán-Cabrera, Fractional mechanical oscillators, Revista mexicana de física 58 (2012) 348–352.
[22] S. Kazem, S. Abbasbandy, S. Kumar, Fractional-order Legendre functions for solving fractional-order differential equations, Appl. Math. Model. 37 (2013) 5498–5510.
[23] H. Jafari, S.A. Yousefi, M.A. Firoozjaeea, S. Momanic, C.M. Khaliqued, Application of legendre wavelets for solving fractional differential equations, Comput. Math. Appl. 62 (2011) 1038–1045.
[24] S. Jain, P. Agarwal, On applications of fractional calculus involving summations of series, Applied Mathematics & Approximation Theory, Ankara, Turkey, 96, 2012. May 17–20
[25] A. Jafariana, S. Measoomya, S. Abbasbandy, Artificial neural networks based modeling for solving Volterra integral equations system, Appl. Soft Comput. 27 (2015) 391–398.
[26] A. Jafariana, M. Mokhtarpour, D. Baleanu, Artificial neural network approach for a class of fractional ordinary differential equation, Neural Comput. Appl. doi:10.1007/s00521-015-2104-8.
[27] I.E. Lagaris, A. Likas, D.I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Trans. Neural Netw. 9 (5) (Sept. 1998) 987–1000.
[28] X. Li, Numerical solution of fractional differential equations using cubic b-spline wavelet collocation method, Commun. Nonlinear Sci. Numer. Simul. 17 (2012) 3934–3946.
[29] D.G. Luenberger, Y. Ye, Linear and Nonlinear Programming, Second ed., Addison-Wesley Publishing Company, 1984.
[30] R. Metzler, J. Klafter, The random walk's guide to anomalous diffusion: a fractional dynamics approach, Phys. Rep. 339 (2000) 1–77.
[31] S. Momani, Z. Odibat, Analytical solution of a time-fractional Navier-Stokes equation by Adomian decomposition method, Appl. Math. Comput. 177 (2006) 488–494.
[32] S. Momani, Z. Odibat, Numerical approach to differential equations of fractional order, J. Comput. Appl. Math. 177 (2007) 96–110.
[33] M. Mosleh, M. Otadi, Simulation and evaluation of fuzzy differential equations by fuzzy neural network, Appl. Soft Comput. 12 (2012) 2817–2827.

[34] Z.M. Odibat, S. Momani, An algorithm for the numerical solution of differential equations of fractional order, J. Appl. Math. Inform. 26 (2008) 15–27.
[35] Z.M. Odibat, S. Momani, Application of variational iteration method to equations of fractional order, Int. J. Nonlinear Sci. Numer. Simul. 7 (2006) 271–279.
[36] I. Podlubny, Fractional Differential Equations, Academic Press, San Diego, CA, 1999.
[37] W. Press, S. Flannery, S. Teukolsky, W. Vetterling, Numerical Recipes: The Art of Scientific Computing, Cambridge University Press, New York, 1986.
[38] A. Saadatmandia, M. Dehghan, A new operational matrix for solving fractional-order differential equations, Comput. Math. Appl. 59 (2010) 1326–1336.
[39] J. Sabouri, S. Effati, M. Pakdaman, A neural network approach for solving a class of fractional optimal control problems, Neural Process. Lett. doi:10.1007/s11063-016-9510-5.
[40] S. Salahshour, T. Allahviranloo, S. Abbasbandy, Solving fuzzy fractional differential equations by fuzzy laplace transforms, Commun. Nonlinear Sci. Numer. Simul. 17 (2012) 1372–1381.
[41] S. Salahshour, A. Ahmadian, N. Senu, D. Baleanu, P. Agarwal, On analytical solutions of the fractional differential equation with uncertainty: application to the basset problem, Entropy 17 (2015) 885–902.
[42] C. Saloma, Computational complexity and observation of physical signals, J. Appl. Phys. 74 (1993) 531–549.
[43] N.H. Sweilam, M.M. Khader, R.F. Al-Bar, Numerical studies for a multi-order fractional differential equation, Phys. Lett. A 371 (2007) 26–33.
[44] D. Valério, J.T. Machado, V. Kiryakova, Historical survey: some pioneers of the applications of fractional calculus, Fract. Calc. Appl. Anal. 17 (2014) 552–578.
[45] M.A.Z. Raja, J.A. Khan, I.M. Qureshi, Evolutionary computational intelligence in solving the fractional differential equations, in: N.T. Nguyen, M.T. Le, J. Światek (Eds.), ACIIDS 2010, Part I, LNAI 5990, 2010, pp. 231–240.
[46] M.A.Z. Raja, J.A. Khan, I.M. Qureshi, Solution of fractional order system of Bagley-Torvik equation using evolutionary computational intelligence, Math. Probl. Eng. 2011 (2011) 675075.
[47] M.A.Z. Raja, M.A. Manzar, R. Samar, An efficient computational intelligence approach for solving fractional order Riccati equations using ANN and SQP, Appl. Math. Model. 39 (2015) 3075–3093.
[48] A. Kilbas, A. Aleksandrovich, H.M. Srivastava, J.J. Trujillo, Theory and Applications of Fractional Differential Equations, vol. 204, Elsevier Science Limited, 2006.
[49] R. Garrappa, On linear stability of predictor-corrector algorithms for fractional differential equations, Int. J. Comput. Math. 87 (2010) 2281–2290.
[50] S. Mall, S. Chakraverty, Application of Legendre neural network for solving ordinary differential equations, Appl. Soft Comput. 43 (2016) 347–356.
[51] J.C. Butcher, The Numerical Analysis of Ordinary Differential Equations: Runge–Kutta and General Linear Methods, Wiley-Interscience, 1987.