Hindawi Publishing Corporation
Advances in Mathematical Physics
Volume 2015, Article ID 439526, 12 pages
http://dx.doi.org/10.1155/2015/439526

Research Article

A Numerical Method for Solving Fractional Differential Equations by Using Neural Network

Haidong Qu[1] and Xuan Liu[2]

[1]*Department of Mathematics, Hanshan Normal University, Chaozhou 521041, China*
[2]*Department of Basic Education, Hanshan Normal University, Chaozhou 521041, China*

Correspondence should be addressed to Haidong Qu; qhaidong@163.com and Xuan Liu; czliuxuan@126.com

Received 5 February 2015; Revised 27 April 2015; Accepted 27 April 2015

Academic Editor: Fawang Liu

We present a new method for solving the fractional differential equations of initial value problems by using neural networks which are constructed from cosine basis functions with adjustable parameters. By training the neural networks repeatedly the numerical solutions for the fractional differential equations were obtained. Moreover, the technique is still applicable for the coupled differential equations of fractional order. The computer graphics and numerical solutions show that the proposed method is very effective.

1. Introduction

Recently, fractional differential equations have gained considerable importance due to their frequent appearance applications in fluid flow, rheology, dynamical processes in self-similar and porous structures, diffusive transport akin to diffusion, electrical networks, probability and statistics, control theory of dynamical systems, viscoelasticity, electrochemistry of corrosion, chemical physics, optics and signal processing [1–7], and so on. These applications in interdisciplinary sciences motivate us to try to find out the analytic or numerical solutions for the fractional differential equations. But for most ones it is difficult to find out or even have exact solutions. Thus, necessarily, the numerical techniques are applied to the fractional differential equations.

Now, many effective methods for solving fractional differential equations have been presented, such as nonlinear functional analysis method including monotone iterative technique [8, 9], topological degree theory [10], and fixed point theorems [11–13]. Also, numerical solutions are obtained by the following methods: random walk [2], matrix approach [14], the Adomian decomposition method and variational iteration method [15], HAM [16–19], homotopy perturbation method (HPM) [20], and so forth. Not long ago, in [21], Raja et al. by applying Particle Swarm Optimization (PSO) algorithm along with feedforward ANN obtained the numerical solutions for fractional differential equations. But the convergence of the algorithm has not been proven, and this method is only applied to the single fractional differential equations. In this paper, we construct two different neural networks based on cosine functions and obtain the conditions of algorithm convergence.

The first neural network (NU) is applied to linear and nonlinear fractional differential equations of the form

$$D_{0+}^{\alpha} y(x) = f(x, y(x)), \quad 0 < x \le 1,\ 0 < \alpha \le 1 \tag{1}$$

with initial condition as follows:

$$y(0) = C, \tag{2}$$

where D_{0+}^{α} is the Caputo fractional derivatives of order α.

The second neural network (NU) is applied to the fractional coupled differential equations of the form

$$\begin{aligned} D_{0+}^{\alpha} y(x) &= f(x, y(x), z(x)), \\ D_{0+}^{\alpha} z(x) &= g(x, y(x), z(x)), \end{aligned} \qquad 0 < x \le 1,\ 0 < \alpha \le 1, \tag{3}$$

with initial conditions as follows:

$$\begin{aligned} y(0) &= C_1, \\ z(0) &= C_2, \end{aligned} \tag{4}$$

where D_{0+}^{α} is the Caputo fractional derivatives of order α. The solutions for the above two problems are written as cosine basis functions, whose parameters can be adjusted to minimize an appropriate error function. So we need to compute the gradient of the error with respect to the network parameters. By adjusting the parameters repeatedly, we obtain the numerical solutions when the error values are less than the required accuracy or the training times reach maximum.

2. Definitions and Lemma

Definition 1 (see [22]). The Riemann-Liouville fractional integral of order $\alpha \in R, \alpha > 0$ of a function $f(x) \in C_{\mu}, \mu \geq -1$ is defined as

$$\left(I_{0+}^{\alpha} f(t)\right)(x) := \frac{1}{\Gamma(\alpha)} \int_0^x \frac{f(t)\,dt}{(x-t)^{1-\alpha}}, \quad (x > 0). \tag{5}$$

Definition 2 (see [22]). The Riemann-Liouville and Caputo fractional derivatives of order $\alpha \in R, \alpha > 0$ are given by

$$\begin{aligned} {}^{\mathrm{RL}}D_{0+}^{\alpha} f(x) &:= \left(\frac{d}{dx}\right)^n I_{0+}^{n-\alpha} f(x) \\ &= \frac{1}{\Gamma(n-\alpha)} \left(\frac{d}{dx}\right)^n \int_0^x \frac{f(t)\,dt}{(x-t)^{\alpha-n+1}}, \\ D_{0+}^{\alpha} f(x) &:= I_{0+}^{n-\alpha} \left(\frac{d}{dx}\right)^n f(x) \\ &= \frac{1}{\Gamma(n-\alpha)} \int_0^x \frac{(d/dt)^n f(t)\,dt}{(x-t)^{\alpha-n+1}}, \end{aligned} \tag{6}$$

where $(n = [\alpha] + 1,\ x > 0)$.

Definition 3 (see [22]). The classical Mittag-Leffler function is defined by

$$\mathrm{E}_{\alpha}(x) := \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(\alpha k + 1)}, \quad (x \in C,\ \alpha > 0). \tag{7}$$

The generalized Mittag-Leffler function is defined by

$$\mathrm{E}_{\alpha,\beta}(x) := \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(\alpha k + \beta)}, \quad (x, \beta \in C,\ \alpha > 0). \tag{8}$$

Definition 4. The functions $\mathrm{Sin}_{\alpha,\beta}(x)$, $\mathrm{Cos}_{\alpha,\beta}(x)$ $(x, \beta \in C,\ \alpha > 0)$ are defined by

$$\begin{aligned} \mathrm{Sin}_{\alpha,\beta}(x) &= \sum_{k=1}^{\infty} (-1)^{k+1} \frac{x^{2k-1}}{\Gamma(\alpha(2k-1) + \beta)}, \\ \mathrm{Cos}_{\alpha,\beta}(x) &= \sum_{k=0}^{\infty} (-1)^{k} \frac{x^{2k}}{\Gamma(\alpha(2k) + \beta)}. \end{aligned} \tag{9}$$

Obviously, Euler's equations have the following forms:

$$\begin{aligned} \mathrm{E}_{\alpha,\beta}(ix) &= \mathrm{Cos}_{\alpha,\beta}(x) + i\mathrm{Sin}_{\alpha,\beta}(x), \\ \mathrm{E}_{\alpha,\beta}(-ix) &= \mathrm{Cos}_{\alpha,\beta}(x) - i\mathrm{Sin}_{\alpha,\beta}(x). \end{aligned} \tag{10}$$

Lemma 5. *If* $\mathrm{Sin}_{\alpha,\beta}(x)$ *and* $\mathrm{Cos}_{\alpha,\beta}(x)$ *are defined as in Definition 4, then*

$$\begin{aligned} &D_{a+}^{\alpha} (x-a)^{\beta-1} \mathrm{Sin}_{\mu,\beta}\left[\lambda (x-a)^{\mu}\right] \\ &\quad = (x-a)^{\beta-\alpha-1} \mathrm{Sin}_{\mu,\beta-\alpha}\left[\lambda (x-a)^{\mu}\right], \end{aligned} \tag{11}$$

$$\begin{aligned} &D_{a+}^{\alpha} (x-a)^{\beta-1} \mathrm{Cos}_{\mu,\beta}\left[\lambda (x-a)^{\mu}\right] \\ &\quad = (x-a)^{\beta-\alpha-1} \mathrm{Cos}_{\mu,\beta-\alpha}\left[\lambda (x-a)^{\mu}\right]. \end{aligned} \tag{12}$$

Proof. The beta function was defined by $\widetilde{\beta}(p,q) := \int_0^1 x^{p-1}(1-x)^{q-1}dx$, and we have the following equation:

$$\widetilde{\beta}(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}. \tag{13}$$

Then according to the definition of Caputo fractional derivatives, we have

$$\begin{aligned} D_{a+}^{\alpha} (x-a)^{\beta-1} \mathrm{Sin}_{\mu,\beta}\left[\lambda (x-a)^{\mu}\right] &= \frac{1}{\Gamma(n-\alpha)} \int_{a+}^{x} \frac{(d/dt)^n (t-a)^{\beta-1} \mathrm{Sin}_{\mu,\beta}\left[\lambda (t-a)^{\mu}\right]}{(x-t)^{\alpha-n+1}} dt \\ &= \frac{1}{\Gamma(n-\alpha)} \int_{0+}^{x-a} \frac{(d/d\xi)^n \xi^{\beta-1} \mathrm{Sin}_{\mu,\beta}\left[\lambda \xi^{\mu}\right]}{(x-\xi-a)^{\alpha-n+1}} d\xi \quad (\xi = t-a) \\ &= \frac{1}{\Gamma(n-\alpha)} \int_{0+}^{1} \frac{(d/dt)^n t^{\beta-1} (x-a)^{\beta-1} \mathrm{Sin}_{\mu,\beta}\left[\lambda t^{\mu} (x-a)^{\mu}\right]}{(x-a)^{n} (x-a)^{\alpha-n+1} (1-t)^{\alpha-n+1}} (x-a)\,dt \quad (\xi = t(x-a)) \end{aligned}$$

$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)} \int_{0+}^{1} \frac{(d/dt)^n t^{\beta-1} \sum_{k=1}^{\infty} \left((-1)^{k+1} \left[\lambda t^{\mu} (x-a)^{\mu}\right]^{2k-1} / \Gamma\left(\mu(2k-1)+\beta\right)\right)}{(1-t)^{\alpha-n+1}} dt$$

$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)} \frac{\sum_{k=1}^{\infty} (-1)^{k+1} \left[\lambda (x-a)^{\mu}\right]^{2k-1}}{\Gamma\left(\mu(2k-1)+\beta\right)} \int_{0+}^{1} \frac{(d/dt)^n t^{\beta-1+\mu(2k-1)}}{(1-t)^{\alpha-n+1}} dt$$

$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)} \frac{\sum_{k=1}^{\infty} (-1)^{k+1} \left[\lambda (x-a)^{\mu}\right]^{2k-1}}{\Gamma\left(\mu(2k-1)+\beta\right)} \prod_{i=1}^{n} \left[\beta - i + \mu(2k-1)\right] \int_{0+}^{1} \frac{t^{\beta-1+\mu(2k-1)-n}}{(1-t)^{\alpha-n+1}} dt$$

$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)} \frac{\sum_{k=1}^{\infty} (-1)^{k+1} \left[\lambda (x-a)^{\mu}\right]^{2k-1}}{\Gamma\left(\mu(2k-1)+\beta-n\right)} \widetilde{\beta}\left(\beta+\mu(2k-1)-n, n-\alpha\right)$$

$$= \frac{(x-a)^{\beta-\alpha-1}}{\Gamma(n-\alpha)} \frac{\sum_{k=1}^{\infty} (-1)^{k+1} \left[\lambda (x-a)^{\mu}\right]^{2k-1}}{\Gamma\left(\mu(2k-1)+\beta-n\right)} \frac{\Gamma\left(\beta+\mu(2k-1)-n\right)\Gamma(n-\alpha)}{\Gamma\left(\beta+\mu(2k-1)-\alpha\right)}$$

$$= (x-a)^{\beta-\alpha-1} \sum_{k=1}^{\infty} \frac{(-1)^{k+1} \left[\lambda (x-a)^{\mu}\right]^{2k-1}}{\Gamma\left(\mu(2k-1)+\beta-\alpha\right)} = (x-a)^{\beta-\alpha-1} \operatorname{Sin}_{\mu,\beta-\alpha}\left[\lambda (x-a)^{\mu}\right]. \tag{14}$$

Then (11) holds. Similarly, we obtain (12). In particular, when $\beta = 1$, $\mu = 1$, we have

$$\begin{aligned} &D_{a+}^{\alpha} \operatorname{Sin}_{1,1}\left[\lambda (x-a)\right] \\ &\quad = D_{a+}^{\alpha} \sin\left[\lambda (x-a)\right] \\ &\quad = (x-a)^{-\alpha} \operatorname{Sin}_{1,1-\alpha}\left[\lambda (x-a)\right], \\ &D_{a+}^{\alpha} \operatorname{Cos}_{1,1}\left[\lambda (x-a)\right] \\ &\quad = D_{a+}^{\alpha} \cos\left[\lambda (x-a)\right] \\ &\quad = (x-a)^{-\alpha} \operatorname{Cos}_{1,1-\alpha}\left[\lambda (x-a)\right]. \end{aligned} \tag{15}$$

□

3. Illustration of the Method and Application

3.1. The First Neural Network. To describe the method, we consider (1) with initial condition $y(0) = C$. The jth trial solution satisfying the initial condition is written as

$$\begin{aligned} y_j(x) = &\sum_{i=1}^{\mathscr{M}} w_{i,j} \cos(ix) \\ &+ \left(C - \sum_{i=1}^{\mathscr{M}} w_{i,j}\right) \cos\left((\mathscr{M}+1)x\right), \end{aligned} \tag{16}$$

where $\mathscr{M}$ represents the number of neurons and $w_{i,j}$ are unknown weights of the network determined in training procedures to reduce the error function:

$$J = \frac{1}{2} \left\| E_j \right\|_2^2 = \frac{1}{2} \sum_{k=1}^{\mathscr{N}} \left(e_j(k)\right)^2, \qquad E_j = \left(e_j(1), e_j(2), \ldots, e_j(\mathscr{N})\right)^T, \tag{17}$$

where $\mathscr{N}$ represents the number of sample points, $\| \cdot \|_2$ is Euclidean norm, and

$$\begin{aligned} e_j(k) &= f\left(x_k, y_j(x_k)\right) - D_{0+}^{\alpha} y_j(x_k) \\ &= f\left(x_k, y_j(x_k)\right) - x_k^{-\alpha} \left(\sum_{i=1}^{\mathscr{M}} w_{i,j} \operatorname{Cos}_{1,1-\alpha}(ix_k) \right. \\ &\quad \left. + \left(C - \sum_{i=1}^{\mathscr{M}} w_{i,j}\right) \operatorname{Cos}_{1,1-\alpha}\left((\mathscr{M}+1)x_k\right) \right), \end{aligned} \tag{18}$$

where $k = 1, 2, \ldots, \mathscr{N}$; then we can adjust the weights $w_{i,j}$ by the following equation:

$$w_{i,j+1} = w_{i,j} + \Delta w_{i,j}, \tag{19}$$

where

$$\begin{aligned} \Delta w_{i,j} &= -\mu \frac{\partial J}{\partial w_{i,j}} = -\mu \sum_{k=1}^{\mathscr{N}} \frac{\partial J}{\partial e_j(k)} \frac{\partial e_j(k)}{\partial w_{i,j}} \\ &= -\mu \sum_{k=1}^{\mathscr{N}} e_j(k) f_y\left(x_k, y_j(x_k)\right) \\ &\quad \cdot \left(\cos(x_k) - \cos\left((\mathscr{M}+1)x_k\right)\right) - (x_k)^{-\alpha} e_j(k) \\ &\quad \cdot \left(\operatorname{Cos}_{1,1-\alpha} x_k - \operatorname{Cos}_{1,1-\alpha}\left((\mathscr{M}+1)x_k\right)\right). \end{aligned} \tag{20}$$

3.2. Convergence of the Algorithm

Theorem A. *Let μ represent learning rate, let $\mathscr{N}$ represent the number of sample points, and let $\mathscr{M}$ represent the number of neurons: $X = (x_1, x_2, \ldots, x_{\mathscr{N}})$, $\delta < x_i < 1$, $1 \le i \le N$. Suppose $|f_y| \le L_1$, $|\operatorname{Cos}_{1,1-\alpha}(x)| \le L_2$ on the interval $(\delta, 1)$ for $0 < \delta < 1$. (From Figure 10, we see that the function*

$\mathrm{Cos}_{1,1-\alpha}(x)$ *is bounded when* $0 \le \alpha \le 1$.*) Then the neural network is convergent on the interval* $(\delta, 1)$ *when*

$$0 < \mu < \frac{\mathcal{N}+1}{4.\mathcal{M}\mathcal{N}\left(L_1+\delta^{-\alpha}L_2\right)^2}. \tag{21}$$

Proof. Let $W_j = (w_{1j}, w_{2j}, \ldots, w_{\mathcal{M}j})$, and then we denote $y_j(X)$ by

$$y_j(X) = W_j \cdot G - \left(C - W_j \cdot I_1\right)\cos\left((\mathcal{M}+1)X\right), \tag{22}$$

where $G = (\cos(X), \cos(2X), \ldots, \cos(\mathcal{M}X))^T$ and $I_1 = (\overbrace{1, 1, \ldots, 1}^{\mathcal{M}})^T$. Then according to (17), we have

$$\begin{aligned} E_j &= \left(e_j(1), e_j(2), \ldots, e_j(\mathcal{N})\right) \\ &= f\left(X, y_j(X)\right) - D_{0+}^{\alpha} y_j(X) \\ &= f\left(X, y_j(X)\right) \\ &\quad - X^{\alpha}\left(W_j H + \left(C - W_j I_1\right)\mathrm{Cos}_{1,1-\alpha}\left((\mathcal{M}+1)X\right)\right), \end{aligned} \tag{23}$$

where $H = (\mathrm{Cos}_{1,1-\alpha}(X), \mathrm{Cos}_{1,1-\alpha}(2X), \ldots, \mathrm{Cos}_{1,1-\alpha}(\mathcal{M}X))^T$. Then we have

$$\begin{aligned} \frac{\partial E_j}{\partial W_j} &= f_y\left(G - I_1\cos\left((\mathcal{M}+1)X\right)\right) \\ &\quad - X^{\alpha}\left(H - I_1\mathrm{Cos}_{1,1-\alpha}\left((\mathcal{M}+1)X\right)\right). \end{aligned} \tag{24}$$

Noting $J = (1/2)\|E_j\|_2^2$, then we get

$$\Delta W_j = -\mu\frac{\partial J}{\partial E_j}\frac{\partial E_j}{\partial W_j} = -\mu E_j\frac{\partial E_j}{\partial W_j}. \tag{25}$$

Define Lyapunov function $V_j = (1/2)\|E_j\|_2^2$; we have

$$\Delta V_j = \frac{1}{2}\left\|E_{j+1}\right\|_2^2 - \frac{1}{2}\left\|E_j\right\|_2^2. \tag{26}$$

Suppose

$$E_{j+1} = E_j + \Delta E_j = E_j + \left(\frac{\partial E_j}{\partial W_j}\right)^T \Delta W_j, \tag{27}$$

and then in accordance with (25) that yields

$$E_{j+1} = \left(I - \mu\left(\frac{\partial E_j}{\partial W_j}\right)^T\frac{\partial E_j}{\partial W_j}\right)E_j, \tag{28}$$

where

$$I = \begin{pmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{pmatrix}_{\mathcal{N}\times\mathcal{N}}. \tag{29}$$

Thus,

$$\begin{aligned} \Delta V_j &= \frac{1}{2}\left\|\left(I - \mu\left(\frac{\partial E_j}{\partial W_j}\right)^T\frac{\partial E_j}{\partial W_j}\right)E_j\right\|_2^2 - \frac{1}{2}\left\|E_j\right\|_2^2 \\ &\le \frac{1}{2}\left(\left\|\left(I - \mu\left(\frac{\partial E_j}{\partial W_j}\right)^T\frac{\partial E_j}{\partial W_j}\right)\right\|_F^2 - 1\right)\left\|E_j\right\|_2^2, \end{aligned} \tag{30}$$

where $\|\cdot\|_F$ is Frobenius matrix norm, defined by $\|(a_{ij})_{n\times n}\|_F = (\sum_{i=1}^n\sum_{j=1}^n a_{ij}^2)^{1/2}$. Since $\|E_j\|_2^2 > 0$, in order to make this neural network converge, we have

$$\left\|I - \mu\left(\frac{\partial E_j}{\partial W_j}\right)^T\frac{\partial E_j}{\partial W_j}\right\|_F^2 < 1, \tag{31}$$

which yields

$$\left\|I - \mu\left(\frac{\partial E_j}{\partial W_j}\right)^T\frac{\partial E_j}{\partial W_j}\right\|_F < 1. \tag{32}$$

Hence,

$$\begin{aligned} 1 &> \left\|I - \mu\left(\frac{\partial E_j}{\partial W_j}\right)^T\frac{\partial E_j}{\partial W_j}\right\|_F \ge \mu\left\|\frac{\partial E_j}{\partial W_j}\right\|_F^2 - \|I\|_F \\ &= \mu\left\|\frac{\partial E_j}{\partial W_j}\right\|_F^2 - \mathcal{N}. \end{aligned} \tag{33}$$

In accordance with $0 < \mu < 1$, we obtain

$$0 < \mu < \frac{\mathcal{N}+1}{\left\|\partial E_j/\partial W_j\right\|_F^2}. \tag{34}$$

By calculating (25), we get

$$\begin{aligned} &\left\|\frac{\partial E_j}{\partial W_j}\right\|_F^2 \\ &\le \left\|\begin{pmatrix} 2\left(L_1+\delta^{-\alpha}L_2\right) & \cdots & 2\left(L_1+\delta^{-\alpha}L_2\right) \\ \vdots & & \vdots \\ 2\left(L_1+\delta^{-\alpha}L_2\right) & \cdots & 2\left(L_1+\delta^{-\alpha}L_2\right) \end{pmatrix}_{\mathcal{M}\times\mathcal{N}}\right\|_F^2 \\ &= 4.\mathcal{M}\mathcal{N}\left(L_1+\delta^{-\alpha}L_2\right)^2. \end{aligned} \tag{35}$$

Finally, we have

$$0 < \mu < \frac{\mathcal{N}+1}{4.\mathcal{M}\mathcal{N}\left(L_1+\delta^{-\alpha}L_2\right)^2}. \tag{36}$$

□

Table 1: Weights ($\times 10^{-4}$) obtained along with the solution of Examples 1, 2, and 3.

α	Example 1			Example 2			Example 3		
	1	0.7	0.5	1	0.7	0.5	1	0.7	0.5
w_1	4857	6310	6665	8226	9941	8880	5415	4070	5101
w_2	−0506	−3466	−4424	4080	0219	2914	−1589	0680	−1582
w_3	−4434	−2721	−1362	−3468	−0621	−3593	−5771	−5800	−3972
w_4	−3170	−3110	−4990	1680	1372	3715	−1309	−2908	−4142
w_5	1896	4204	5290	−1455	−1468	−3147	3295	3947	6096
w_6	5534	−0926	0222	1895	0482	1372	2815	2901	−0121
w_7	−6316	0182	−2151	−1350	0424	0273	−4609	−4203	−2015

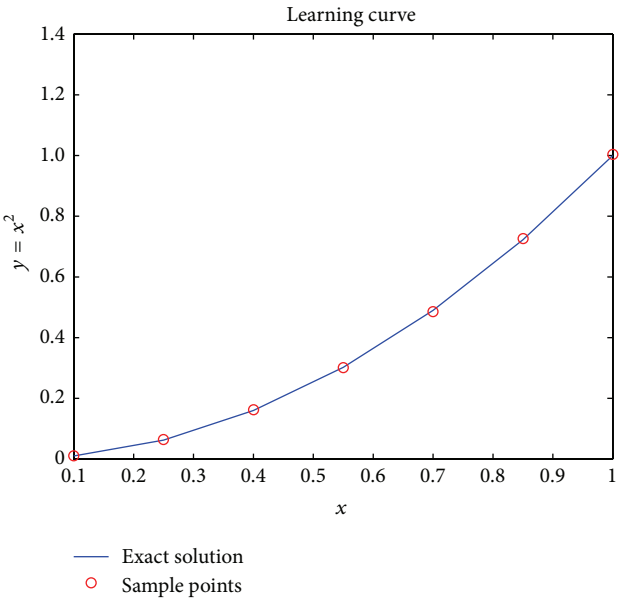

Figure 1: The learning curve for Example 1.

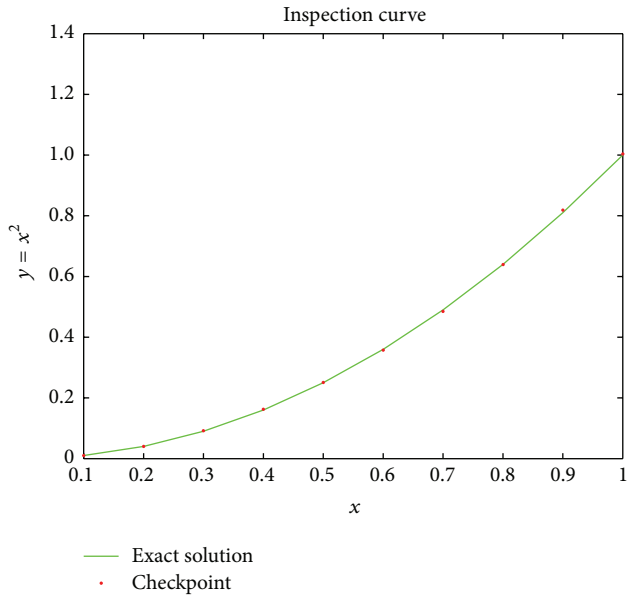

Figure 2: The inspection curve for Example 1.

3.3. Example

3.3.1. Example 1. We first consider the following linear fractional differential equation:

$$D_{0+}^{\alpha} y(x) = x^2 + \frac{2}{\Gamma(3-\alpha)} x^{2-\alpha} - y(x), \tag{37}$$

with condition $y(0) = 0$. The exact solution is $y(x) = x^2$. This equation also can be solved by the following methods: Genetic Algorithm (GA) [21], Grünwald-Letnikov classical numerical technique (GL) [23], and Particle Swarm Optimization (PSO) algorithm [23]. We set the parameters $\mu = 0.001$, $\mathscr{M} = 7$, and $\mathscr{N} = 10$ and train the neural network 4500 times, and the weights of the network for Example 1 are given in Table 1. Figure 1 shows that sample points are on the exact solution curve after training is completed. Then we check whether the other points also match well with the exact solution (see Figure 2). From Figure 3 we see the error values decrease rapidly. Tables 2(a) and 2(b) show the numerical solutions and accuracy for Example 1 by the different methods. In this paper, all numerical experiments are done by using Lenovo T400, Intel Core 2 Duo CPU P8700, 2.53 GHz, and Matlab version R2010b. The neural networks with cosine basis functions have taken about 850 s, but the other algorithms mentioned above need to run about 2,240 s.

3.3.2. Example 2. We secondly consider the following linear fractional differential equation:

$$D_{0+}^{\alpha} y(x) = \cos(x) + x^{-\alpha} \mathrm{Cos}_{1,1-\alpha}(x) - y(x), \tag{38}$$

with condition $y(0) = 1$. The exact solution is $y(x) = \cos(x)$. We set the parameters $\mu = 0.001$, $\mathscr{M} = 7$, and $\mathscr{N} = 10$ and train the neural network 1000 times, and the weights of the network for Example 2 are given in Table 1. Figures 4, 5, and 6 show that the neural network is still applicable when $C \neq 0$. Table 3 shows the exact solution, approximate solution, and accuracy for Example 2.

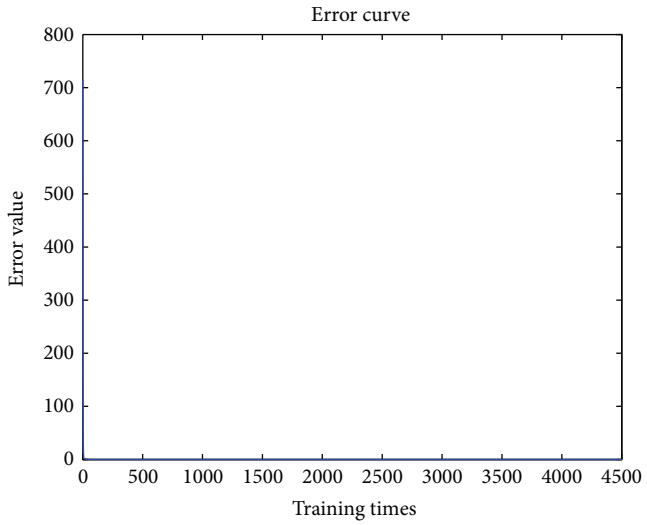

Figure 3: The error curve for Example 1.

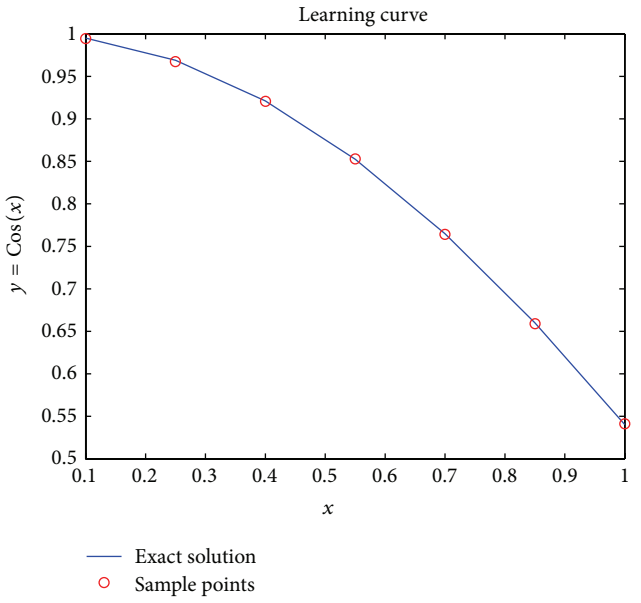

Figure 4: The learning curve for Example 2.

3.3.3. Example 3. We thirdly consider the following nonlinear fractional differential equation:

$$D_{0+}^{\alpha} y(x) = x^{6} + \frac{\Gamma(3.5)}{\Gamma(3.5-\alpha)} x^{2.5-\alpha} - x y^{2}(x), \tag{39}$$

with condition $y(0) = 0$. The exact solution is $y(x) = x^{5/2}$. We set the parameters $\mu = 0.001$, $\mathscr{M} = 7$, and $\mathscr{N} = 10$ and train the neural network 1000 times, and the weights of the network for Example 2 are given in Table 1. Table 4 shows the exact solution, approximate solution, and accuracy for Example 3.

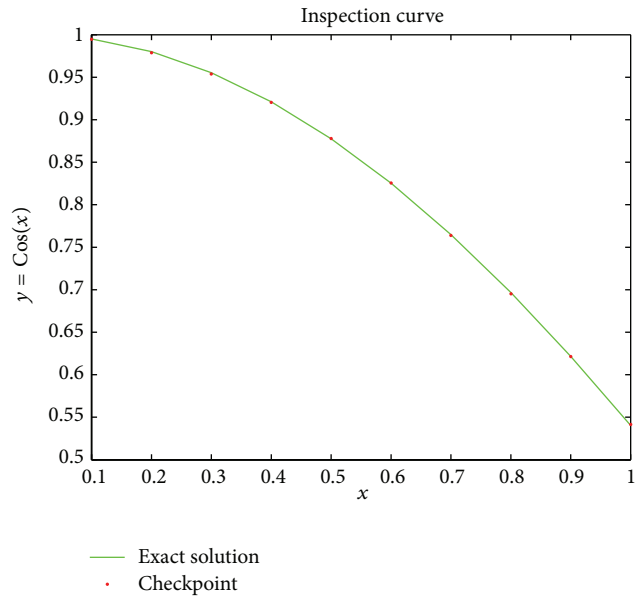

Figure 5: The inspection curve for Example 2.

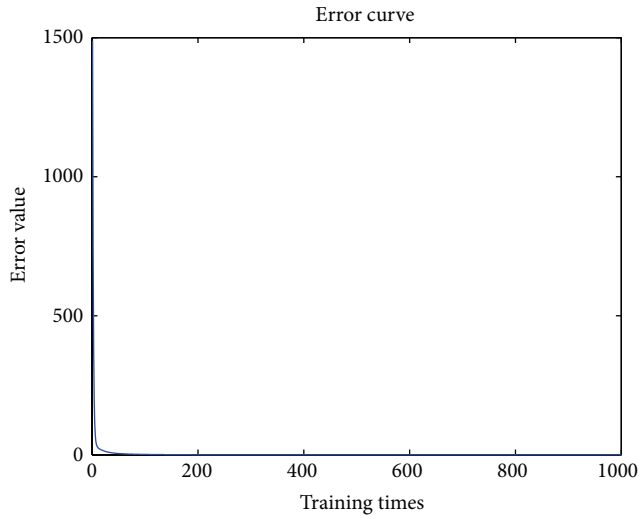

Figure 6: The error curve for Example 2.

3.4. The Second Neural Network. To describe the method, we consider (3) with initial conditions $y(0) = C_1$ and $z(0) = C_2$. The jth trial solutions for the problem are written as

$$\begin{aligned} y_j(x) &= \sum_{i=1}^{\mathscr{M}} w_{i,j} \cos(ix) \\ &\quad + \left(C_1 - \sum_{i=1}^{\mathscr{M}} w_{i,j}\right) \cos((\mathscr{M}+1)x), \\ z_j(x) &= \sum_{i=1}^{\mathscr{M}} q_{i,j} \cos(ix) \\ &\quad + \left(C_2 - \sum_{i=1}^{\mathscr{M}} q_{i,j}\right) \cos((\mathscr{M}+1)x), \end{aligned} \tag{40}$$

Table 2: (a) Comparison of results for the solution of Example 1 for $\alpha = 0.5$. (b) Comparison of results for the solution of Example 1 for $\alpha = 0.75$.

(a)

x	$y(x)$	Numerical solution				Accuracy			
		GL	PSO	GA	NU	GL	PSO	GA	NU
0.1	0.01				0.0101				10^{-4}
0.2	0.04	0.0401	0.0404	0.0396	0.0407	10^{-4}	10^{-4}	10^{-4}	10^{-4}
0.3	0.09	0.0901	0.0907		0.0917	10^{-4}	10^{-4}		10^{-3}
0.4	0.16	0.1601	0.1604	0.1596	0.1621	10^{-4}	10^{-4}	10^{-4}	10^{-3}
0.5	0.25	0.2501	0.2496		0.2505	10^{-4}	10^{-4}		10^{-4}
0.6	0.36	0.3602	0.3583	0.3573	0.3571	10^{-3}	10^{-3}	10^{-3}	10^{-3}
0.7	0.49	0.4902	0.4869		0.4853	10^{-3}	10^{-3}		10^{-3}
0.8	0.64	0.6402	0.6362	0.6352	0.6397	10^{-3}	10^{-3}	10^{-3}	10^{-4}
0.9	0.81	0.8102	0.8069		0.8186	10^{-3}	10^{-3}		10^{-3}
1	1	0.1001	0.1000	0.1004	0.1003	10^{-4}	10^{-5}	10^{-4}	10^{-3}

(b)

x	$y(x)$	Numerical solution			Accuracy		
		GL	PSO	NU	GL	PSO	NU
0.1	0.01	0.0107	0.0103	0.0092	10^{-4}	10^{-4}	10^{-4}
0.2	0.04	0.0413	0.0414	0.0377	10^{-3}	10^{-3}	10^{-3}
0.3	0.09	0.0918	0.0928	0.0875	10^{-3}	10^{-3}	10^{-3}
0.4	0.16	0.1622	0.1636	0.1592	10^{-3}	10^{-3}	10^{-4}
0.5	0.25	0.2527	0.2538	0.2511	10^{-3}	10^{-3}	10^{-3}
0.6	0.36	0.3631	0.3631	0.3609	10^{-3}	10^{-3}	10^{-4}
0.7	0.49	0.4934	0.4918	0.4884	10^{-3}	10^{-3}	10^{-3}
0.8	0.64	0.6438	0.6402	0.6373	10^{-3}	10^{-4}	10^{-3}
0.9	0.81	0.8141	0.8091	0.8106	10^{-3}	10^{-4}	10^{-4}
1	1	1.0044	0.9991	1.0020	10^{-3}	10^{-4}	10^{-3}

Table 3: Exact solution, approximate solution, and accuracy for Example 2.

	α	Numerical solution			Accuracy		
x	$\cos(x)$	1	0.7	0.5	1	0.7	0.5
0.1	0.9950	0.9945	0.9967	0.9972	10^{-4}	10^{-3}	10^{-3}
0.2	0.9800	0.9788	0.9852	0.9867	10^{-3}	10^{-3}	10^{-3}
0.3	0.9553	0.9538	0.9620	0.9638	10^{-3}	10^{-3}	10^{-3}
0.4	0.9210	0.9203	0.9249	0.9256	10^{-4}	10^{-3}	10^{-3}
0.5	0.8775	0.8777	0.8758	0.8747	10^{-4}	10^{-3}	10^{-3}
0.6	0.8253	0.8254	0.8196	0.8176	10^{-4}	10^{-3}	10^{-3}
0.7	0.7648	0.7639	0.7607	0.7601	10^{-4}	10^{-3}	10^{-3}
0.8	0.6967	0.6951	0.6990	0.7016	10^{-3}	10^{-3}	10^{-3}
0.9	0.6216	0.6213	0.6286	0.6328	10^{-4}	10^{-3}	10^{-2}
1	0.5403	0.5414	0.5414	0.5405	10^{-3}	10^{-3}	10^{-4}

Table 4: Exact solution, approximate solution, and accuracy for Example 3.

	α	Numerical solution			Accuracy		
x	$x^{5/2}$	1	0.7	0.5	1	0.7	0.5
0.1	0.0031	0.0022	0.0055	0.0066	10^{-4}	10^{-3}	10^{-3}
0.2	0.0178	0.0133	0.0234	0.0266	10^{-3}	10^{-3}	10^{-3}
0.3	0.0492	0.0426	0.0566	0.0603	10^{-3}	10^{-3}	10^{-3}
0.4	0.1011	0.0972	0.1075	0.1093	10^{-3}	10^{-3}	10^{-3}
0.5	0.1767	0.1773	0.1783	0.1772	10^{-4}	10^{-3}	10^{-4}
0.6	0.2788	0.2797	0.2733	0.2711	10^{-4}	10^{-3}	10^{-3}
0.7	0.4099	0.4055	0.4010	0.4009	10^{-3}	10^{-3}	10^{-3}
0.8	0.5724	0.5643	0.5712	0.5738	10^{-3}	10^{-3}	10^{-3}
0.9	0.7684	0.7670	0.7832	0.7847	10^{-3}	10^{-2}	10^{-2}
1	1	1.0064	1.0105	1.0056	10^{-3}	10^{-2}	10^{-3}

where $\mathcal{N}$ represents the number of sample points and $w_{i,j}$ and $q_{i,j}$ are unknown weights of the network determined in training procedures to reduce error function:

$$J^{y+z} = \frac{1}{2}\left\|E_j^{y+z}\right\|_2^2 = \frac{1}{2}\sum_{k=1}^{\mathcal{N}}\left(e_j^y(k)\right)^2 + \frac{1}{2}\sum_{k=1}^{\mathcal{N}}\left(e_j^z(k)\right)^2, \quad (41)$$

where

$$E_j^{y+z} = \left(e_j^y(1), e_j^y(2), \ldots, e_j^y(\mathcal{N}), e_j^z(1), e_j^z(2), \ldots, e_j^z(\mathcal{N})\right)^T,$$

$$e_j^y(k) = f\left(x_k, y_j(x_k), z_j(x_k)\right) - D_{0+}^{\alpha} y_j(x_k)$$

$$
\begin{aligned}
&= f\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right) \\
&- x^{-\alpha}\left(\sum_{i=1}^{\mathscr{M}} w_{i,j}\mathrm{Cos}_{1,1-\alpha}\left(ix_k\right) + \left(C_1 - \sum_{i=1}^{\mathscr{M}} w_{i,j}\right)\right. \\
&\left.\cdot \mathrm{Cos}_{1,1-\alpha}\left(\left(\mathscr{M}+1\right)x_k\right)\right), \\
e_j^z\left(k\right) &= g\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right) - D_{0+}^{\alpha} z_j\left(x_k\right) \\
&= g\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right) \\
&- x^{-\alpha}\left(\sum_{i=1}^{\mathscr{M}} q_{i,j}\mathrm{Cos}_{1,1-\alpha}\left(ix_k\right) + \left(C_2 - \sum_{i=1}^{\mathscr{M}} q_{i,j}\right)\right. \\
&\left.\cdot \mathrm{Cos}_{1,1-\alpha}\left(\left(\mathscr{M}+1\right)x_k\right)\right).
\end{aligned} \tag{42}
$$

Then we adjust the weights $w_{i,j}$ and $q_{i,j}$ by the following two equations:

$$
\begin{aligned}
w_{i,j+1} &= w_{i,j} + \Delta w_{i,j}, \\
q_{i,j+1} &= q_{i,j} + \Delta q_{i,j},
\end{aligned} \tag{43}
$$

where

$$
\begin{aligned}
\Delta w_{i,j} &= -\mu\frac{\partial J^{y+z}}{\partial w_{i,j}} = -\mu\left(\sum_{k=1}^{\mathscr{N}} \frac{\partial J^{y+z}}{\partial e_j^y\left(k\right)}\frac{\partial e_j^y\left(k\right)}{\partial w_{i,j}}\right. \\
&\left. + \sum_{k=1}^{\mathscr{N}} \frac{\partial J^{y+z}}{\partial e_j^z\left(k\right)}\frac{\partial e_j^z\left(k\right)}{\partial w_{i,j}}\right) = -\mu\sum_{k=1}^{\mathscr{N}} e_j^y\left(k\right) \\
&\cdot \left(f_y\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right)\right. \\
&\cdot \left(\cos\left(x_k\right) - \cos\left(\left(\mathscr{M}+1\right)x_k\right)\right) - \left(x_k\right)^{(-\alpha)} \\
&\left.\cdot \left(\mathrm{Cos}_{1,1-\alpha}x_k - \mathrm{Cos}_{1,1-\alpha}\left(\left(\mathscr{M}+1\right)x_k\right)\right)\right) \\
&- \mu\sum_{k=1}^{\mathscr{N}} e_j^z\left(k\right) g_y\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right) \\
&\cdot \left(\cos\left(x_k\right) - \cos\left(\left(\mathscr{M}+1\right)x_k\right)\right), \\
\Delta q_{i,j} &= -\mu\frac{\partial J^{y+z}}{\partial q_{i,j}} = -\mu\left(\sum_{k=1}^{\mathscr{N}} \frac{\partial J^{y+z}}{\partial e_j^z\left(k\right)}\frac{\partial e_j^z\left(k\right)}{\partial q_{i,j}}\right. \\
&\left. + \sum_{k=1}^{\mathscr{N}} \frac{\partial J^{y+z}}{\partial e_j^y\left(k\right)}\frac{\partial e_j^y\left(k\right)}{\partial q_{i,j}}\right) = -\mu\sum_{k=1}^{\mathscr{N}} e_j^z\left(k\right) \\
&\cdot \left(g_z\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right)\right. \\
&\cdot \left(\cos\left(x_k\right) - \cos\left(\left(\mathscr{M}+1\right)x_k\right)\right) - \left(x_k\right)^{(-\alpha)} \\
&\left.\cdot \left(\mathrm{Cos}_{1,1-\alpha}x_k - \mathrm{Cos}_{1,1-\alpha}\left(\left(\mathscr{M}+1\right)x_k\right)\right)\right) \\
&- \mu\sum_{k=1}^{\mathscr{N}} e_j^y\left(k\right) f_z\left(x_k, y_j\left(x_k\right), z_j\left(x_k\right)\right)\left(\cos\left(x_k\right)\right. \\
&\left. - \cos\left(\left(\mathscr{M}+1\right)x_k\right)\right).
\end{aligned} \tag{44}
$$

3.5. Convergence of the Algorithm

Theorem B. *Let μ represent learning rate, let $\mathscr{N}$ represent the number of sample points, and let $\mathscr{M}$ represent the number of neurons: $X = (x_1, x_2, \ldots, x_{\mathscr{N}})$, $\delta < x_i < 1$, $1 \le i \le \mathscr{N}$. Suppose $|f_y| \le L_1^y$, $|f_z| \le L_1^z$, $|g_y| \le L_3^y$, $|g_z| \le L_3^z$, $|\mathrm{Cos}_{1,1-\alpha}(x)| \le L_2$ on the interval $(\delta, 1)$ for $0 < \delta < 1$. Then the neural network is convergent on the interval $(\delta, 1)$ when*

$$
\begin{aligned}
&0 \\
&< \mu \\
&< \frac{2\mathscr{N}+1}{4\mathscr{M}\mathscr{N}\left(\left(L_1^y + L_3^y + \delta^{-\alpha}L_2\right)^2 + \left(L_1^z + L_3^z + \delta^{-\alpha}L_2\right)^2\right)}.
\end{aligned} \tag{45}
$$

Proof. Let $W_j^{y+z} = (w_{1j}, w_{2j}, \ldots, w_{\mathscr{M}j}, q_{1j}, q_{2j}, \ldots, q_{\mathscr{M}j})$, and then we denote $y_j(X)$ and $z_j(X)$ by

$$
\begin{aligned}
y_j\left(X\right) &= W_j^{y+z}G^{1,0} \\
&+ \left(C_1 - W_j^{y+z}I_1^{1,0}\right)\cos\left(\left(\mathscr{M}+1\right)X\right),
\end{aligned} \tag{46}
$$

$$
\begin{aligned}
z_j\left(X\right) &= W_j^{y+z}G^{0,1} \\
&+ \left(C_2 - W_j^{y+z}I_1^{0,1}\right)\cos\left(\left(\mathscr{M}+1\right)X\right),
\end{aligned} \tag{47}
$$

respectively, where

$$
\begin{aligned}
E_j^{y+z} &= \left(e_j^y\left(1\right), e_j^y\left(2\right), \ldots, e_j^y\left(\mathscr{N}\right), e_j^z\left(1\right), e_j^z\left(2\right), \ldots,\right. \\
&\left. e_j^z\left(\mathscr{N}\right)\right) = \left(f\left(X, y_j\left(X\right), z_j\left(X\right)\right) - D_{0+}^{\alpha}y_j\left(X\right),\right. \\
&\left. g\left(X, y_j\left(X\right), z_j\left(X\right)\right) - D_{0+}^{\alpha}z_j\left(X\right)\right) = f\left(X,\right. \\
&\left. y_j\left(X\right), z_j\left(X\right)\right) - X^{\alpha}\left(W_j^{y+z}H^{1,0}\right. \\
&\left. + \left(C_1 - W_j^{y+z}I_1^{1,0}\right)\mathrm{Cos}_{1,1-\alpha}\left(\left(\mathscr{M}+1\right)X\right)\right) + g\left(X,\right. \\
&\left. y_j\left(X\right), z_j\left(X\right)\right) - X^{\alpha}\left(W_j^{y+z}H^{0,1}\right. \\
&\left. + \left(C_2 - W_j^{y+z}I_1^{0,1}\right)\mathrm{Cos}_{1,1-\alpha}\left(\left(\mathscr{M}+1\right)X\right)\right),
\end{aligned}
$$

$$H^{1,0} = \left(\mathrm{Cos}_{1,1-\alpha}(X), \mathrm{Cos}_{1,1-\alpha}(2X), \ldots, \mathrm{Cos}_{1,1-\alpha}(\mathscr{M}X), \overbrace{0,0,\ldots,0}^{\mathscr{M}} \right)^T,$$

$$H^{0,1} = \left(\overbrace{0,0,\ldots,0}^{\mathscr{M}}, \mathrm{Cos}_{1,1-\alpha}(X), \mathrm{Cos}_{1,1-\alpha}(2X), \ldots, \mathrm{Cos}_{1,1-\alpha}(\mathscr{M}X) \right)^T,$$

$$G^{1,0} = \left(\cos(X), \cos(2X), \ldots, \cos(\mathscr{M}X), \overbrace{0,0,\ldots,0}^{\mathscr{M}} \right)^T,$$

$$G^{0,1} = \left(\overbrace{0,0,\ldots,0}^{\mathscr{M}}, \cos(X), \cos(2X), \ldots, \cos(\mathscr{M}X) \right)^T,$$

$$I_1^{1,0} = \left(\overbrace{1,1,\ldots,1}^{\mathscr{M}}, \overbrace{0,0,\ldots,0}^{\mathscr{M}} \right)^T,$$

$$I_1^{0,1} = \left(\overbrace{0,0,\ldots,0}^{\mathscr{M}}, \overbrace{1,1,\ldots,1}^{\mathscr{M}} \right)^T. \tag{48}$$

Then we have

$$\begin{aligned} \frac{\partial E_j^{y+z}}{\partial W_j^{y+z}} &= f_y\left(G^{1,0} - I_1^{1,0}\cos\left((\mathscr{M}+1)X\right)\right) \\ &\quad + f_z\left(G^{0,1} - I_1^{0,1}\cos\left((\mathscr{M}+1)X\right)\right) \\ &\quad - X^{\alpha}\left(H^{1,0} - I_1^{1,0}\mathrm{Cos}_{1,1-\alpha}\left((\mathscr{M}+1)X\right)\right) \\ &\quad + g_z\left(G^{0,1} - I_1^{0,1}\cos\left((\mathscr{M}+1)X\right)\right) \\ &\quad + g_y\left(G^{1,0} - I_1^{1,0}\cos\left((\mathscr{M}+1)X\right)\right) \\ &\quad - X^{\alpha}\left(H^{0,1} - I_1^{0,1}\mathrm{Cos}_{1,1-\alpha}\left((\mathscr{M}+1)X\right)\right). \end{aligned} \tag{49}$$

Define Lyapunov function $V_j^{y+z} = (1/2)\|E_j^{y+z}\|_2^2$; then similarly to the proof of Theorem A we get

$$0 < \mu < \frac{2\mathscr{N}+1}{\left\|\partial E_j^{y+z}/\partial W_j^{y+z}\right\|_F^2}. \tag{50}$$

By simply calculating $\partial E_j^{y+z}/\partial W_j^{y+z}$, we have

$$\begin{aligned} \left\|\frac{\partial E_j^{y+z}}{\partial W_j^{y+z}}\right\|_F^2 &\le \left\| 2\begin{pmatrix} \mathscr{A} & \cdots & \mathscr{A} & \mathscr{B} & \cdots & \mathscr{B} \\ \vdots & & \vdots & \vdots & & \vdots \\ \mathscr{A} & \cdots & \mathscr{A} & \mathscr{B} & \cdots & \mathscr{B} \end{pmatrix}_{\mathscr{M}\times 2\mathscr{N}} \right\|_F^2 \\ &= 4\mathscr{M}\mathscr{N}\left(\mathscr{A}^2 + \mathscr{B}^2\right), \end{aligned} \tag{51}$$

where $\mathscr{A} = L_1^y + L_3^y + \delta^{-\alpha}L_2$ and $\mathscr{B} = L_1^z + L_3^z + \delta^{-\alpha}L_2$, and finally we obtain

$$\begin{aligned} &0 \\ &< \mu \\ &< \frac{2\mathscr{N}+1}{4\mathscr{M}\mathscr{N}\left(\left(L_1^y + L_3^y + \delta^{-\alpha}L_2\right)^2 + \left(L_1^z + L_3^z + \delta^{-\alpha}L_2\right)^2\right)}. \end{aligned} \tag{52}$$

This completes the proof. □

3.6. Example

3.6.1. Example 4. We first consider the following linear coupled fractional differential equations:

$$\begin{aligned} D_{0+}^{\alpha} y(x) &= x^2 + x^3 + \frac{2}{\Gamma(3-\alpha)}x^{2-\alpha} - y(x) - z(x), \\ D_{0+}^{\alpha} z(x) &= x^2 + x^3 + \frac{6}{\Gamma(4-\alpha)}x^{3-\alpha} - y(x) - z(x), \end{aligned} \quad 0 < x \le 1,\ 0 < \alpha \le 1, \tag{53}$$

with initial condition as follows:

$$\begin{aligned} y(0) &= 0, \\ z(0) &= 0. \end{aligned} \tag{54}$$

The exact solution is $y(x) = x^2$ and $z(x) = x^3$. We set the parameters $\alpha = 0.9$, $\mu = 0.001$, $\mathscr{M} = 7$, and $\mathscr{N} = 10$ and train the neural network 2000 times, and the weights of the network for Example 4 are given in Table 5. Figures 7 and 8 show that the sample points and checkpoints are in well agreement with the exact solutions for the problem. Figure 9 shows that the error of the numerical solutions decreases rapidly within the first 50 training times. Table 6 shows the exact solution, approximate solution, and accuracy for Example 4.

Table 5: Weights obtained along with the solution of Examples 4 and 5.

$\alpha = 0.9$	Example 4	Example 5
w_1/q_1	0.5307/0.4494	0.8215/0.7682
w_2/q_2	−0.0592/0.0013	0.3545/ − 0.7084
w_3/q_3	−0.6414/ − 0.7451	−0.1512/ − 0.2292
w_4/q_4	0.0052/ − 0.0392	−0.1342/ − 0.1116
w_5/q_5	0.0559/0.3705	0.1657/0.3582
w_6/q_6	0.3998/0.2282	−0.0815/0.0781
w_7/q_7	−0.4141/ − 0.4014	0.0533/ − 0.2259

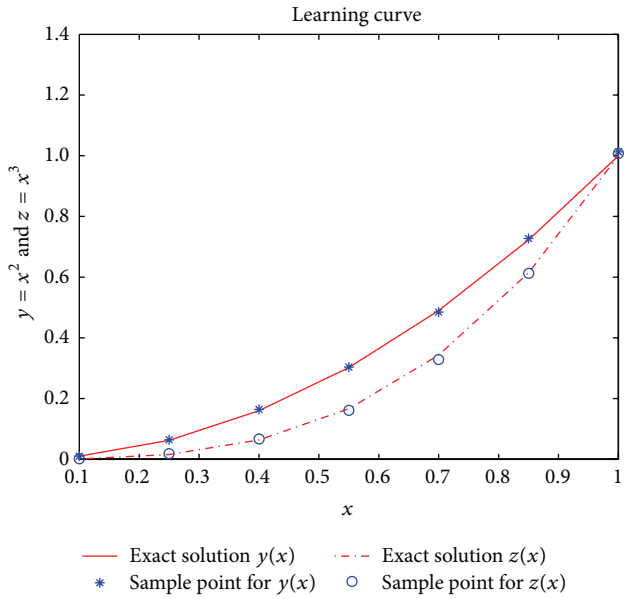

Figure 7: The learning curve for Example 4.

3.6.2. Example 5. We second consider the following nonlinear fractional coupled differential equations:

$$D_{0+}^{\alpha} y(x) = x^6 + \cos(x) + \frac{6}{\Gamma(4-\alpha)} x^{3-\alpha} - (y(x))^2 - z(x), \quad 0 < x \le 1,\ 0 < \alpha \le 1,$$
$$D_{0+}^{\alpha} z(x) = x^3 + \cos(x) + x^{-\alpha} \mathrm{Cos}_{1,1-\alpha}(x) - y(x) - z(x), \tag{55}$$

with initial conditions as follows:

$$y(0) = 0, \quad z(0) = 1, \tag{56}$$

The exact solution is $y(x) = x^3$ and $z(x) = \cos(x)$. We set the parameters $\alpha = 0.9$, $\mu = 0.001$, $\mathscr{M} = 7$, and $\mathscr{N} = 10$. Numerical solutions in Table 7 show that this network can also be applied to the nonlinear fractional coupled differential equations but we need more time to train the network.

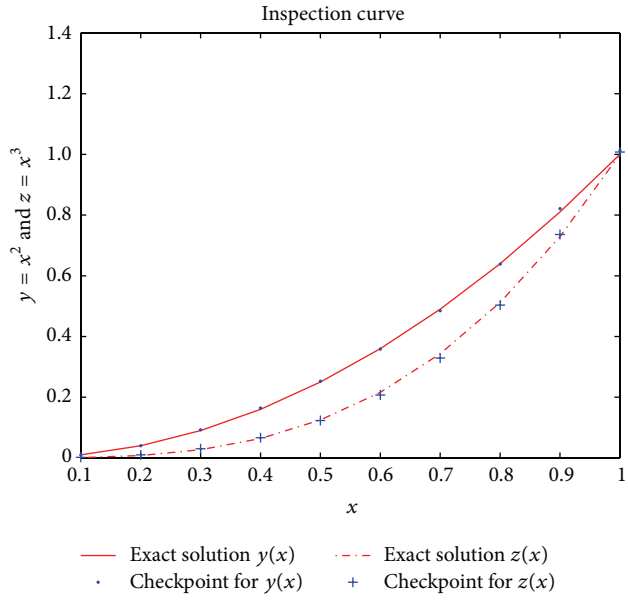

Figure 8: The inspection curve for Example 4.

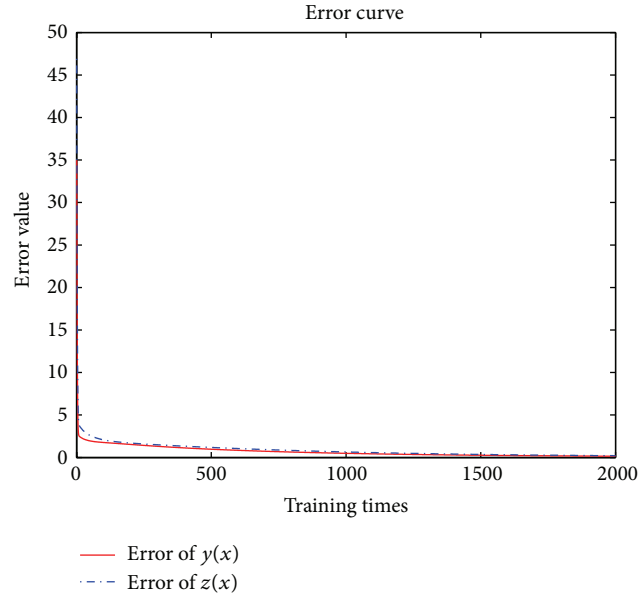

Figure 9: The error curve for Example 4.

4. Conclusion

In this paper, by using the neural network, we obtained the numerical solutions for single fractional differential equations and the systems of coupled differential equations of fractional order. The computer graphics demonstrates that numerical results are in well agreement with the exact solutions. In (1), suppose that $f(x, y(x)) = A(x) + B(x)y + C(x)y^2$; then the problem transformed into Fractional Riccati Equations (Example 3 in this paper). In (3), suppose that $f(x, y(x)) = y(x)(r - ay(x) - bz(x))$ and $g(x, z(x)) = z(x)(-d + cy(x))$; then the problem transformed into

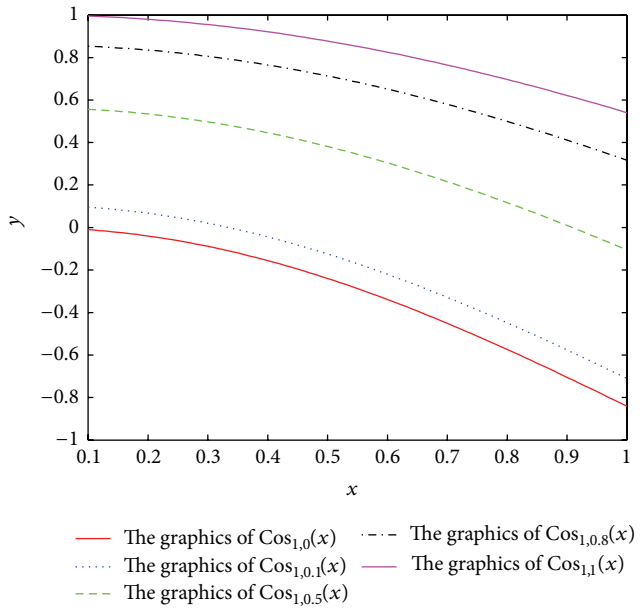

Figure 10: The graphics of the function $\mathrm{Cos}_{\alpha,1-\alpha}(x)$.

Table 6: Exact solution, approximate solution, and accuracy for Example 4.

	$\alpha = 0.9$		Numerical solution		Accuracy	
x	$y(x)$	$z(x)$	$y(x)$	$z(x)$	$y(x)$	$z(x)$
0.1	0.01	0.001	0.0101	0.0020	10^{-4}	10^{-3}
0.2	0.04	0.008	0.0408	0.0104	10^{-4}	10^{-3}
0.3	0.09	0.027	0.0926	0.0301	10^{-3}	10^{-3}
0.4	0.16	0.064	0.1641	0.0664	10^{-3}	10^{-3}
0.5	0.25	0.125	0.2529	0.1233	10^{-3}	10^{-3}
0.6	0.36	0.216	0.3584	0.2071	10^{-3}	10^{-3}
0.7	0.49	0.343	0.4847	0.3290	10^{-3}	10^{-2}
0.8	0.64	0.512	0.6389	0.5033	10^{-3}	10^{-3}
0.9	0.81	0.729	0.8215	0.7361	10^{-2}	10^{-3}
1	1	1	1.0126	1.0076	10^{-2}	10^{-3}

fractional-order Lotka-Volterra predator-prey system. We will consider this problem in another paper. The neural network is a powerful method and is effective for the above two problems, which should be also able to solve fractional partial differential equations.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to thank the referees for their many constructive comments and suggestions to improve the paper. This work was partly supported by the Special Funds of the National Natural Science Foundations of China under Grant no. 11247310, the Foundations for Distinguished Young Talents in Higher Education of Guangdong under Grant no. 2012LYM0096, and the Fund of Hanshan Normal University under Grant nos. LY201302 and LF201403.

Table 7: Exact solution, approximate solution, and accuracy for Example 5.

	$\alpha = 0.9$		Numerical solution		Accuracy	
x	$y(x)$	$z(x)$	$y(x)$	$z(x)$	$y(x)$	$z(x)$
0.1	0.9950	0.001	0.9926	0.0042	10^{-3}	10^{-3}
0.2	0.9800	0.008	0.9834	0.0148	10^{-3}	10^{-3}
0.3	0.9553	0.027	0.9625	0.0351	10^{-3}	10^{-3}
0.4	0.9210	0.064	0.9267	0.0689	10^{-3}	10^{-3}
0.5	0.8775	0.125	0.8773	0.1220	10^{-4}	10^{-3}
0.6	0.8253	0.216	0.8198	0.2040	10^{-3}	10^{-3}
0.7	0.7648	0.343	0.7604	0.3276	10^{-3}	10^{-2}
0.8	0.6967	0.512	0.7004	0.5049	10^{-3}	10^{-3}
0.9	0.6216	0.729	0.6336	0.7375	10^{-2}	10^{-3}
1	0.5403	1	0.5475	1.0056	10^{-3}	10^{-3}

References

[1] A. I. Saichev and G. M. Zaslavsky, "Fractional kinetic equations: solutions and applications," *Chaos*, vol. 7, no. 4, pp. 753–764, 1997.

[2] R. Metzler and J. Klafter, "The random walk's guide to anomalous diffusion: a fractional dynamics approach," *Physics Reports*, vol. 339, no. 1, pp. 1–77, 2000.

[3] R. Metzler and J. Klafter, "Boundary value problems for fractional diffusion equations," *Physica A: Statistical Mechanics and its Applications*, vol. 278, no. 1-2, pp. 107–125, 2000.

[4] S. B. Yuste, L. Acedo, and K. Lindenberg, "Reaction front in an $A + \vec{B}C$ reaction-subdiffusion process," *Physical Review E*, vol. 69, no. 3, Article ID 036126, 2004.

[5] K. Diethelm and A. D. Freed, "On the solution of nonlinear fractional order differential equations used in the modeling of viscoplasticity," in *Scientifice Computing in Chemical Engineering II—Computational Fluid Dynamics, Reaction Engineering and Molecular Properties*, F. Keil, W. Mackens, H. Voss, and J. Werther, Eds., pp. 217–224, Springer, Heidelberg, Germany, 1999.

[6] L. Gaul, P. Klein, and S. Kemple, "Damping description involving fractional operators," *Mechanical Systems and Signal Processing*, vol. 5, no. 2, pp. 81–88, 1991.

[7] Z. Odibat and S. Momani, "Numerical methods for nonlinear partial differential equations of fractional order," *Applied Mathematical Modelling*, vol. 32, no. 1, pp. 28–39, 2008.

[8] Z. Liu and J. Liang, "A class of boundary value problems for first-order impulsive integro-differential equations with deviating arguments," *Journal of Computational and Applied Mathematics*, vol. 237, no. 1, pp. 477–486, 2013.

[9] Z. H. Liu, J. H. Sun, and I. Szántó, "Monotone iterative technique for Riemann-Liouville fractional integro-differential equations with advanced arguments," *Results in Mathematics*, vol. 63, no. 3-4, pp. 1277–1287, 2013.

[10] Z. H. Liu, N. Van Loi, and V. Obukhovskii, "Existence and global bifurcation of periodic solutions to a class of differential variational inequalities," *International Journal of Bifurcation and Chaos*, vol. 23, no. 7, Article ID 1350125, 2013.

[11] Z. Liu, L. Lu, and I. Szántó, "Existence of solutions for fractional impulsive differential equations with p-Laplacian operator," *Acta Mathematica Hungarica*, vol. 141, no. 3, pp. 203–219, 2013.

[12] X. Liu and Z. Liu, "Existence results for a class of second order evolution inclusions and its corresponding first order evolution inclusions," *Israel Journal of Mathematics*, vol. 194, no. 2, pp. 723–743, 2013.

[13] H. D. Qu and X. Liu, "Existence of nonnegative solutions for a fractional m-point boundary value problem at resonance," *Boundary Value Problems*, vol. 2013, article 127, 2013.

[14] I. Podlubny, A. Chechkin, T. Skovranek, Y. Chen, and B. M. Vinagre Jara, "Matrix approach to discrete fractional calculus. II. Partial fractional differential equations," *Journal of Computational Physics*, vol. 228, no. 8, pp. 3137–3153, 2009.

[15] Z. Odibat, S. Momani, and H. Xu, "A reliable algorithm of homotopy analysis method for solving nonlinear fractional differential equations," *Applied Mathematical Modelling*, vol. 34, no. 3, pp. 593–600, 2010.

[16] S. Das and P. K. Gupta, "Homotopy analysis method for solving fractional hyperbolic partial differential equations," *International Journal of Computer Mathematics*, vol. 88, no. 3, pp. 578–588, 2011.

[17] A. Elsaid, "Homotopy analysis method for solving a class of fractional partial differential equations," *Communications in Nonlinear Science and Numerical Simulation*, vol. 16, no. 9, pp. 3655–3664, 2011.

[18] L. Song and H. Zhang, "Solving the fractional BBM-Burgers equation using the homotopy analysis method," *Chaos, Solitons & Fractals*, vol. 40, no. 4, pp. 1616–1622, 2009.

[19] H. Jafari, A. Golbabai, S. Seifi, and K. Sayevand, "Homotopy analysis method for solving multi-term linear and nonlinear diffusion-wave equations of fractional order," *Computers & Mathematics with Applications*, vol. 59, no. 3, pp. 1337–1344, 2010.

[20] S. Momani and Z. Odibat, "Homotopy perturbation method for nonlinear partial differential equations of fractional order," *Physics Letters. A*, vol. 365, no. 5-6, pp. 345–350, 2007.

[21] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "Evolutionary computation technique for solving Riccati differential equation of arbitrary order," *World Academy of Science, Engineering and Technology*, vol. 58, pp. 531–536, 2009.

[22] A. A. Kilbsa, H. M. Srivastava, and J. J. Trujillo, *Theory and Applications of Fractional Differential Equations*, Elsevier, Amsterdam, The Netherlands, 2006.

[23] M. A. Z. Raja, J. A. Khan, and I. M. Qureshi, "A new stochastic approach for solution of Riccati differential equation of fractional order," *Annals of Mathematics and Artificial Intelligence*, vol. 60, no. 3-4, pp. 229–250, 2010.