Solution of dual fuzzy equations using a new iterative method

Sina Razvarz [1], Raheleh Jafari [2], Ole-Christoffer Granmo[2], Alexander Gegov[3]

[1] Departamento de Control Automatico CINVESTAV-IPN (National Polytechnic Institute)
Mexico City, Mexico
srazvarz@yahoo.com
[2] Department of Information and Communication Technology, Agder University College, 4876 Grimstad, Norway
Jafari3339@yahoo.com
ole.granmo@uia.no
[3] School of Computing, University of Portsmouth, Buckingham Building, Portsmouth PO13HE, UK
alexander.gegov@port.ac.uk

Abstract. In this paper, a new hybrid scheme based on learning algorithm of fuzzy neural network (FNN) is offered in order to extract the approximate solution of fully fuzzy dual polynomials (FFDPs). Our FNN in this paper is a five-layer feed-back FNN with the identity activation function. The input-output relation of each unit is defined by the extension principle of Zadeh. The output from this neural network, which is also a fuzzy number, is numerically compared with the target output. The comparison of the feed-back FNN method with the feed-forward FNN method shows that the less error is observed in the feed-back FNN method. An example based on applications are given to illustrate the concepts, which are discussed in this paper.

Keywords: Fully Fuzzy Dual Polynomials, Fuzzy Neural Network, Approximate Solution.

1 Introduction

Artificial neural networks (ANNs) are mathematical or computational models based on biological neural networks. They make effort to imitate the information presentation, processing scheme and discrimination capability of natural neurons in the human brain. ANNs are a prominent component of artificial intelligence, which emulate the learning procedure of the human brain for extracting patterns from historical data. Neural networks can be categorized as feed-forward and feed-back ones. The primary futileness of feed-forward neural networks is that the weight updating does not employ any information on the local data structure, also the function approximation is impressionable to the training data [1]. However, feed-back neural networks have impressive representation abilities so that can successfully overcome the futileness of

feed-forward neural networks. In recent years, there have been a wide spread of studies in the field of neural networks [2-6]. Ishibuchi et al. [7] designed a FNN with triangular fuzzy weights. Abbasbandy et al. [8] investigated the solution of polynomials like $a_1x+...+a_nx^n=a_0$ where $x\in\Re$ and $a_0,a_1,\ldots,a_n$ are fuzzy numbers by a learning algorithm of FNN. Jafarian et al. [9] proposed a new learning algorithm for solving fuzzy polynomials. Friedman et al. [10] represented a new model for solving a fuzzy $n\times n$ linear system with crisp coefficient matrix and a fuzzy vector in the right-hand side. Also they investigated the duality of the fuzzy linear systems like $Ax=Bx+y$ where A and B are two real $n\times n$ matrices and the unknown x and y are two vectors with n fuzzy numbers components [11]. In [12] a FNN model is utilized in order to extract the coefficients of fuzzy polynomial regression.

Up to now, however, none has been reported on applications of the FNNs to solve FFDP. This kind of polynomial has been widely studied due to its promising potential for applications in different fields such as engineering, physics, economics and optimal control theory. Thus, in this paper the FNN is a first and important step for solving these polynomials. In [13], the authors have been proposed Newton's method for solving fuzzy nonlinear equations. Dehghan et al. [14] introduced a numerical method to solve a system of linear fuzzy equation. The solution of fuzzy polynomial equation based on the ranking method has been investigated by [15]. In [16] the ranking technique is implemented in order to obtain the real roots of dual fuzzy polynomial equation. Muzzioli et al. [17] applied nonlinear programming method for the solution of fuzzy linear system. Amirfakhrian in [18] presented a numerical iterative method to find the roots of an algebraic fuzzy equation of degree n with fuzzy coefficients. In [19] the fully fuzzy system of linear equations with an arbitrary fuzzy coefficient is investigated. Ezzati [20] developed a new method for solving an arbitrary general fuzzy linear system by using the embedding approach. In [21] solving fully fuzzy system of linear equations by using multi objective linear programming and the embedding approach is discussed. Waziri et al. [22] applied a new approach for solving dual fuzzy nonlinear equations by using Broyden's and Newton's methods. Also, in [23] the Adomian decomposition method for solving these polynomials is introduced. In [24] the exponent to production technique is illustrated in order to generate an analytical and approximated solution of fully fuzzy quadratic equation. Babbar et al. [25] have applied a new approach to find the nonnegative solution of a fully fuzzy linear system, where the elements of the coefficient matrix are defined as arbitrary triangular fuzzy numbers. More information on fuzzy polynomials can be found in [26, 27].

The objective of this paper is to design a new model based on FNNs for approximate solution of FFDPs. In this work, a model of feed-back FNN equivalent to dual fuzzy polynomial of the form $a_1x+\cdots+a_nx^n=b_1x+\cdots+b_nx^n+d$ is built, where a_j,b_j,d and x are fuzzy numbers (for $j=1,\ldots,n$). The input-output relation of each unit of the designed neural network is defined by the extension principle of Zadeh [28]. The proposed feed-back FNN is able to estimate the fuzzy solution related to FFDP to any level of preciseness. In continues, by comparing our results with the results obtained by using feed-forward FNN, it can be observed that the feed-back method yields faster convergence rate and less computational complexity in the ad-

justing the weights. This paper is started with a brief description of FFDPs in Section 2. In this section, the feed-back FNN and feed-forward FNN are introduced. Furthermore, by using the learning algorithm which is derived from the cost function, we will be capable of finding a fuzzy solution associated with the FFDP. An example is likewise presented in section 3. Section 4 finishes the paper with conclusions.

2 Fully fuzzy dual polynomials

In this part the interest is vested in finding a solution for the following FFDP

$$p_1 y + \cdots + p_n y^n = q_1 y + \cdots + q_n y^n + \Phi \tag{1}$$

where $p_i, q_i, \Phi, \mathrm{y} \in \mathrm{E}(\mathrm{i} = 1, \ldots, \mathrm{n})$. In order to extract an estimated solution associated with the FFDP two models of feed-back FNN and feed-forward FNN equivalent to Eq. (1) are presented in Figure 1 and Figure 2 respectively.

Generally, for an arbitrary fuzzy number $a \in E$, there exists no element $b \in E$ such that, $a + b = 0$. In fact, for all non-crisp fuzzy number $a \in E$ we have $a + (-a) \neq 0$. Hence, Eq. (8) cannot be equivalently substituted by $(p_1 - q_1)y + \cdots + (p_n - q_n)y^n = \Phi$, which had been investigated.

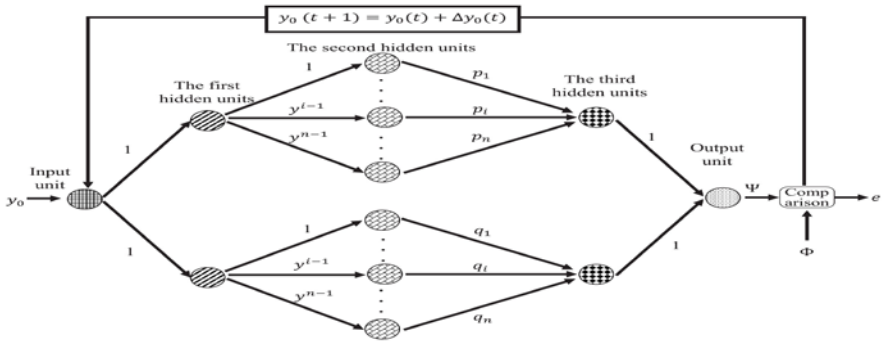

Fig. 1. Feed-back FNN for resolving dual fuzzy polynomials

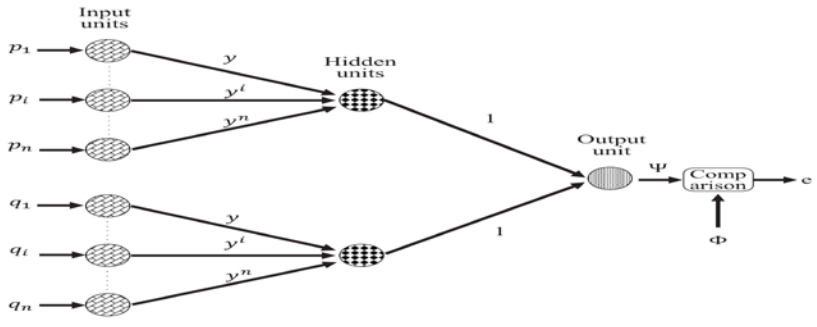

Fig. 2. Feed-forward FNN for resolving dual fuzzy polynomials

2.1 Computation of fuzzy output in feed-back FNNs

In the presented neural network training data are written as $\{(P, Q); \Phi\}$ where $P = (p_1, \dots, p_n)$ and $Q = (q_1, \dots, q_n)$. The α-level sets associated with the fuzzy coefficients p_i as well as q_i are nonnegative, i.e., $0 \le \underline{p}_i^{\alpha} \le \overline{p}_i^{\alpha}$ and $0 \le \underline{q}_i^{\alpha} \le \overline{q}_i^{\alpha}$ for all i's. We have

- Input unit

$$[y_0]^{\alpha} = (\underline{y}_0^{\alpha}, \overline{y}_0^{\alpha}) \tag{2}$$

- The first hidden units

$$[Z_{11}]^{\alpha} = \left(\underline{y}_0^{\alpha}, \overline{y}_0^{\alpha}\right), [Z_{12}]^{\alpha} = (\underline{y}_0^{\alpha}, \overline{y}_0^{\alpha}) \tag{3}$$

- The second hidden units

$$[Z_{21}]^{\alpha} = (\sum_{i\varepsilon M_{21}} \left(\underline{y}_0^{\alpha}\right)^i + \sum_{i\varepsilon C_{21}} (\overline{y}_0^{\alpha})^i + \sum_{i\varepsilon N_{21}} \left(\underline{y}_0^{\alpha}\right)^i, \sum_{i\varepsilon M_{22}} (\overline{y}_0^{\alpha})^i + \sum_{i\varepsilon C_{22}} \left(\underline{y}_0^{\alpha}\right)^i + \sum_{i\varepsilon N_{22}} (\overline{y}_0^{\alpha})^i) \tag{4}$$

$$[Z_{22}]^{\alpha} = (\sum_{i\varepsilon M_{21}} \left(\underline{y}_0^{\alpha}\right)^i + \sum_{i\varepsilon C_{21}} (\overline{y}_0^{\alpha})^i + \sum_{i\varepsilon N_{21}} \left(\underline{y}_0^{\alpha}\right)^i, \sum_{i\varepsilon M_{22}} (\overline{y}_0^{\alpha})^i + \sum_{i\varepsilon C_{22}} \left(\underline{y}_0^{\alpha}\right)^i + \sum_{i\varepsilon N_{22}} (\overline{y}_0^{\alpha})^i) \tag{5}$$

Where $M_{21} = \left\{i \left| \underline{y}_0^{\alpha} \ge 0 \right.\right\}$, $C_{21} = \left\{i \left| \underline{y}_0^{\alpha} < 0 \right., i \text{ is even number}\right\}$, $N_{21} = \{i \left| \underline{y}_0^{\alpha} < 0 \right.,$ $i \text{ is odd number}\}$, $M_{22} = \{i | \overline{y}_0^{\alpha} \ge 0\}$, $C_{22} = \{i | \overline{y}_0^{\alpha} < 0, i \text{ is even number}\}$,$N_{22} = \{i | \overline{y}_0^{\alpha} < 0,$ $i \text{ is odd number}\}$, $M_{21} \cup C_{21} \cup N_{21} = \{1, \dots, n\}$ and $M_{22} \cup C_{22} \cup N_{22} = \{1, \dots, n\}$

- The third hidden units

$$[Z_{31}]^{\alpha} = (\sum_{j\varepsilon M_{31}} \underline{Z}_{21}^{\alpha} \underline{p}_i^{\alpha} + \sum_{j\varepsilon C_{31}} \underline{Z}_{21}^{\alpha} \overline{p}_i^{\alpha}, \sum_{j\varepsilon M'_{31}} \overline{Z}_{21}^{\alpha} \overline{p}_i^{\alpha} + \sum_{j\varepsilon C'_{31}} \overline{Z}_{21}^{\alpha} \underline{P}_i^{\alpha}) \tag{6}$$

where $M_{31} = \{i \mid \underline{Z}_{21}^{\alpha} \ge 0\}$, $C_{31} = \{i \mid \underline{Z}_{21}^{\alpha} < 0\}$, $M'_{31} = \{i \mid \overline{Z}_{21}^{\alpha} \ge 0\}$, $C'_{31} = \{i \mid \overline{Z}_{21}^{\alpha} < 0\}$, $M_{31} \cup C_{31} = \{1,...,n\}$ and $M'_{31} \cup C'_{31} = \{1,...,n\}$.

$$[Z_{32}]^{\alpha} = (-\sum_{i\varepsilon M_{32}} \underline{Z}_{22}^{\alpha} \underline{q}_i^{\alpha} - \sum_{j\varepsilon C_{32}} \underline{Z}_{22}^{\alpha} \overline{q}_i^{\alpha}, -\sum_{j\varepsilon M'_{32}} \overline{Z}_{22}^{\alpha} \overline{q}_i^{\alpha} - \sum_{j\varepsilon C'_{32}} \overline{Z}_{22}^{\alpha} \mathrm{q}_i^{\alpha}) \tag{7}$$

where $M_{32} = \{i \mid \underline{Z}_{22}^{\alpha} \ge 0\}$, $C_{32} = \{j \mid [\underline{Z}_{22}^{\alpha} < 0\}$, $M'_{32} = \{j \mid \overline{Z}_{22}^{\alpha} \ge 0\}$, $C'_{32} = \{j \mid \overline{Z}_{22}^{\alpha} < 0\}$, $M_{32} \cup C_{32} = \{1,...,n\}$ and $M'_{32} \cup C'_{32} = \{1,...,n\}$.

- Output unit

$$[\Psi]^{\alpha} = (\underline{Z}_{31}^{\alpha} + \underline{Z}_{32}^{\alpha}, \overline{Z}_{31}^{\alpha} + \overline{Z}_{32}^{\alpha}) \tag{8}$$

The triangular fuzzy weight y_0 is indicated considering the three parameters mentioned as $y_0 = (y_0^1, y_0^2, y_0^3)$. Taking into account the fuzzy parameter y_0, its adjustment rule has been portrayed as below

$$\begin{gathered} y_0^r(t+1) = y_0^r(t) + \Delta y_0^r(t), \quad r = 1,2,3 \\ \Delta y_0^r(t) = -\eta \frac{\partial e^{\alpha}}{\partial y_0^r} + \gamma \Delta y_0^r(t-1) \end{gathered} \tag{9}$$

where t is referred as the number of adjustments, η is taken to be the learning constant, also γ is referred as a momentum constant. We calculate $\frac{\partial e^{\alpha}}{\partial y_0^r}$ as follows

$$\frac{\partial e^{\alpha}}{\partial y_0^r} = \frac{\partial \underline{e}^{\alpha}}{\partial y_0^r} + \frac{\partial \overline{e}^{\alpha}}{\partial y_0^r} \tag{10}$$

Hence complexities lies in the calculation of the derivatives $\frac{\partial \underline{e}^{\alpha}}{\partial y_0^r}$ and $\frac{\partial \overline{e}^{\alpha}}{\partial y_0^r}$. So we have

$$\frac{\partial \underline{e}^{\alpha}}{\partial y_0^r} = -\alpha(\underline{\Phi}^{\alpha} - \underline{\Psi}^{\alpha})(\frac{\partial \underline{net}_{31}^{\alpha}}{\partial y_0^r} - \frac{\partial \underline{net}_{32}^{\alpha}}{\partial y_0^r}) \tag{11}$$

where

$$\frac{\partial \underline{net}_{31}^{\alpha}}{\partial y_0^r} = \sum_{j \varepsilon M_{31}} \underline{P}_i^{\alpha} \frac{\partial \underline{Z}_{21}^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon C_{31}} \overline{p}_i^{\alpha} \frac{\partial \underline{Z}_{21}^{\alpha}}{\partial y_0^r} \tag{12}$$

$$\frac{\partial \underline{net}_{32}^{\alpha}}{\partial y_0^r} = \sum_{j \varepsilon M_{32}} \underline{q}_i^{\alpha} \frac{\partial \underline{Z}_{22}^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon C_{32}} \overline{q}_i^{\alpha} \frac{\partial \underline{Z}_{22}^{\alpha}}{\partial y_0^r} \tag{13}$$

$$\frac{\partial \underline{Z}_{21}^{\alpha}}{\partial y_0^r} = \frac{\partial \underline{Z}_{22}^{\alpha}}{\partial y_0^r} = \sum_{j \varepsilon M_{21}} i(\underline{y}_0^{\alpha})^{i-1} \frac{\partial \underline{y}_0^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon C_{21}} i(\overline{y}_0^{\alpha})^{i-1} \frac{\partial \underline{y}_0^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon N_{21}} i(\underline{y}_0^{\alpha})^{i-1} \frac{\partial \underline{y}_0^{\alpha}}{\partial y_0^r} \tag{14}$$

and

$$\frac{\partial \overline{e}^{\alpha}}{\partial y_0^r} = -\alpha(\overline{\Phi}^{\alpha} - \overline{\Psi}^{\alpha})(\frac{\partial \overline{net}_{31}^{\alpha}}{\partial y_0^r} - \frac{\partial \overline{net}_{32}^{\alpha}}{\partial y_0^r}) \tag{15}$$

where

$$\frac{\partial \overline{net}_{31}^{\alpha}}{\partial y_0^r} = \sum_{j \varepsilon M'_{31}} \overline{p}_i^{\alpha} \frac{\partial \overline{Z}_{21}^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon C'_{31}} \underline{p}_i^{\alpha} \frac{\partial \overline{Z}_{21}^{\alpha}}{\partial y_0^r} \tag{16}$$

$$\frac{\partial \overline{net}_{32}^{\alpha}}{\partial y_0^r} = \sum_{j \varepsilon M'_{32}} \overline{q}_i^{\alpha} \frac{\partial \overline{Z}_{22}^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon C'_{32}} \underline{q}_i^{\alpha} \frac{\partial \overline{Z}_{22}^{\alpha}}{\partial y_0^r} \tag{17}$$

$$\frac{\partial \overline{Z}_{21}^{\alpha}}{\partial y_0^r} = \frac{\partial \overline{Z}_{22}^{\alpha}}{\partial y_0^r} = \sum_{j \varepsilon M_{22}} i(\overline{y}_0^{\alpha})^{i-1} \frac{\partial \overline{y}_0^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon C_{22}} i(\underline{y}_0^{\alpha})^{i-1} \frac{\partial \underline{y}_0^{\alpha}}{\partial y_0^r} + \sum_{j \varepsilon N_{22}} i(\overline{y}_0^{\alpha})^{i-1} \frac{\partial \overline{y}_0^{\alpha}}{\partial y_0^r} \tag{18}$$

In above relations the derivatives $\frac{\partial \underline{y}_0^{\alpha}}{\partial y_0^r}$ and $\frac{\partial \overline{y}_0^{\alpha}}{\partial y_0^r}$ can be summarized as follows

$$\frac{\partial \underline{y}_0^{\alpha}}{\partial y_0^r} = \begin{cases} 1-\alpha, & r=1 \\ \alpha, & r=2 \\ 0, & r=3 \end{cases} \quad \frac{\partial \overline{y}_0^{\alpha}}{\partial y_0^r} = \begin{cases} 0, & r=1 \\ \alpha, & r=2 \\ 1-\alpha, & r=3 \end{cases} \tag{19}$$

2.2 Computation of fuzzy output in feed-forward FNNs

The α -level sets associated with the fuzzy inputs p_i 's as well as q_i's are nonnegative, i.e., $0 \le \underline{p}_i^{\alpha} \le \overline{p}_i^{\alpha}$ and $0 \le \underline{q}_i^{\alpha} \le \overline{q}_i^{\alpha}$ for all i 's. We have

- Input units

$$[p_i]^{\alpha} = \left(\underline{p}_i^{\alpha}, \overline{p}_i^{\alpha}\right), [q_i]^{\alpha} = \left(\underline{q}_i^{\alpha}, \overline{q}_i^{\alpha}\right), \quad i = 1,2,\dots,n \tag{20}$$

- Hidden units

$$[Z_1]^{\alpha} = \left(\sum_{i\varepsilon M}\left(\underline{y}^{\alpha}\right)^i \underline{p}_i^{\alpha} + \sum_{i\varepsilon C}\left(\underline{y}^{\alpha}\right)^i \overline{p}_i^{\alpha}, \sum_{i\varepsilon M'}\left(\overline{y}^{\alpha}\right)^i \overline{p}_i^{\alpha} + \sum_{i\varepsilon C'}\left(\overline{y}^{\alpha}\right)^i \underline{p}_i^{\alpha}\right) \tag{21}$$

$$[Z_2]^{\alpha} = \left(-\sum_{i\varepsilon M}\left(\underline{y}^{\alpha}\right)^i \underline{q}_i^{\alpha} - \sum_{i\varepsilon C}\left(\underline{y}^{\alpha}\right)^i \overline{q}_i^{\alpha}, -\sum_{i\varepsilon M'}\left(\overline{y}^{\alpha}\right)^i \overline{q}_i^{\alpha} - \sum_{i\varepsilon C'}\left(\overline{y}^{\alpha}\right)^i \underline{q}_i^{\alpha}\right) \tag{22}$$

where $M = \{i \mid (\underline{y}^{\alpha})^i \ge 0\}$, $C = \{i \mid (\underline{y}^{\alpha})^i < 0\}$, $M' = \{i \mid (\overline{y}^{\alpha})^i \ge 0\}$, $C' = \{i \mid (\overline{y}^{\alpha})^i < 0\}$, $M \cup C = \{1,\dots,n\}$ and $M' \cup C' = \{1,\dots,n\}$.

- Output unit

$$[\Psi]^{\alpha} = \left(\underline{Z}_1^{\alpha} + \underline{Z}_2^{\alpha}, \overline{Z}_1^{\alpha} + \overline{Z}_2^{\alpha}\right) \tag{23}$$

Assume Φ to be the fuzzy target output in association with the fuzzy coefficient vectors (p_i, q_i). A cost function which is required to be minimized is stated at par with the α-level sets of the fuzzy output Ψ as well as the target output Φ as $e^{\alpha} = \underline{e}^{\alpha} + \overline{e}^{\alpha}$, where $\underline{e}^{\alpha} = \alpha \frac{(\underline{\Phi}^{\alpha} - \underline{\Psi}^{\alpha})^2}{2}$ and $\overline{e}^{\alpha} = \alpha \frac{(\overline{\Phi}^{\alpha} - \overline{\Psi}^{\alpha})^2}{2}$. The $\underline{e}^{\alpha}$ and $\overline{e}^{\alpha}$ are demonstrated as the squared errors for the lower limits as well as the upper limits associated with the α-level sets of the fuzzy output Ψ and target output Φ, respectively.

The triangular fuzzy weight y is indicated considering the three parameters mentioned as $y = (y^1, y^2, y^3)$. The weight is adjusted by the following rule [7]

$$\begin{aligned} y^r(t+1) &= y^r(t) + \Delta y^r(t), \quad r = 1,2,3 \\ \Delta y^r(t) &= -\eta \frac{\partial e^{\alpha}}{\partial y^r} + \gamma \Delta y^r(t-1) \end{aligned} \tag{24}$$

where t is referred as the number of adjustments, η is taken to be as the learning constant, also γ is referred as a momentum constant. We calculate $\frac{\partial e^{\alpha}}{\partial y^r}$ as follows

$$\frac{\partial e^{\alpha}}{\partial y^{r}} = \frac{\partial \underline{e}^{\alpha}}{\partial y^{r}} + \frac{\partial \overline{e}^{\alpha}}{\partial y^{r}} \tag{25}$$

Hence complexities lies in the calculation of the derivatives $\frac{\partial \underline{e}^{\alpha}}{\partial y^{r}}$ and $\frac{\partial \overline{e}^{\alpha}}{\partial y^{r}}$. So we have

$$\frac{\partial \underline{e}^{\alpha}}{\partial y^{r}} = -\alpha(\underline{\Phi}^{\alpha} - \underline{\Psi}^{\alpha})(\frac{\partial \underline{net}_{21}^{\alpha}}{\partial y^{r}} - \frac{\partial \underline{net}_{22}^{\alpha}}{\partial y^{r}}) \tag{26}$$

where

$$\frac{\partial \underline{net}_{21}^{\alpha}}{\partial y_{0}^{r}} = \sum_{i\varepsilon \mathrm{M}} \underline{p}_{i}^{\alpha} i(\underline{y}^{\alpha})^{i-1} \frac{\partial \underline{y}^{\alpha}}{\partial y^{r}} + \sum_{i\varepsilon \mathrm{C}} \overline{p}_{i}^{\alpha} i(\underline{y}^{\alpha})^{i-1} \frac{\partial \underline{y}^{\alpha}}{\partial y^{r}} \tag{27}$$

$$\frac{\partial \underline{net}_{22}^{\alpha}}{\partial y_{0}^{r}} = \sum_{i\varepsilon \mathrm{M}} \underline{q}_{i}^{\alpha} i(\underline{y}^{\alpha})^{i-1} \frac{\partial \underline{y}^{\alpha}}{\partial y^{r}} + \sum_{i\varepsilon \mathrm{C}} \overline{q}_{i}^{\alpha} i(\underline{y}^{\alpha})^{i-1} \frac{\partial \underline{y}^{\alpha}}{\partial y^{r}} \tag{28}$$

and

$$\frac{\partial \overline{e}^{\alpha}}{\partial y^{r}} = -\alpha(\overline{\Phi}^{\alpha} - \overline{\Psi}^{\alpha})(\frac{\partial \overline{net}_{21}^{\alpha}}{\partial y^{r}} - \frac{\partial \overline{net}_{22}^{\alpha}}{\partial y^{r}}) \tag{29}$$

Where

$$\frac{\partial \overline{net}_{21}^{\alpha}}{\partial y^{r}} = \sum_{i\varepsilon M'} \overline{p}_{i}^{\alpha}\, i(\overline{y}^{\alpha})^{i-1} \frac{\partial \overline{y}^{\alpha}}{\partial y^{r}} + \sum_{i\varepsilon C'} \underline{p}_{i}^{\alpha} i(\overline{y}^{\alpha})^{i-1} \frac{\partial \overline{y}^{\alpha}}{\partial y^{r}} \tag{30}$$

$$\frac{\partial \overline{net}_{22}^{\alpha}}{\partial y^{r}} = \sum_{i\varepsilon M'} \overline{q}_{i}^{\alpha}\, i(\overline{y}^{\alpha})^{i-1} \frac{\partial \overline{y}^{\alpha}}{\partial y^{r}} + \sum_{i\varepsilon C'} \underline{q}_{i}^{\alpha} i(\overline{y}^{\alpha})^{i-1} \frac{\partial \overline{y}^{\alpha}}{\partial y^{r}} \tag{31}$$

In above relations the derivatives $\frac{\partial \underline{y}^{\alpha}}{\partial y^{r}}$ and $\frac{\partial \overline{y}^{\alpha}}{\partial y^{r}}$ can be summarized as follows

$$\frac{\partial \underline{y}^{\alpha}}{\partial y^{r}} = \begin{cases} 1-\alpha, & r=1 \\ \alpha, & r=2 \\ 0, & r=3 \end{cases}, \quad \frac{\partial \overline{y}^{\alpha}}{\partial y^{r}} = \begin{cases} 0, & r=1 \\ \alpha, & r=2 \\ 1-\alpha, & r=3 \end{cases} \tag{32}$$

3 Numerical examples

To show the behavior and properties of the proposed method, an examples is solved. **Example:** A vertical propeller shaft AQ with a diameter of $d = 0.015$ is connected to the fixed base of . The propeller shaft is made of steel with $G = 80 \times 10^{9}$ and the resultant torques in the points A, B, C and D are $T_1 = y, T_2 = y^2, T_3 = y$ and $T_4 = y^2$, respectively. The resultant torques T_1 and T_2 can cause a twisting in shaft which is equal to φ degree, see Fig. 4a. The resultant torques T_3 and T_4 can cause a twisting in shaft which is equal to $\varphi \ominus (5,9,14)$ degree, see Fig. 3b. According to the torsion equation we will have [29]:

$$\varphi = \frac{L_1 T_1}{JG} \oplus \frac{L_2 T_2}{JG} = \frac{L_3 T_3}{JG} \oplus \frac{L_4 T_4}{JG} \oplus (5,9,14) \tag{33}$$

where

$$J = \frac{\Pi}{2} d^4. \tag{34}$$

The length of the shafts are not exact, which satisfy the triangular function (1),

$$\begin{aligned} L_1 &= (7,8,11) = p_1 \\ L_2 &= (1,2,3) = p_2 \\ L_3 &= (1,2,3) = q_1 \\ L_4 &= (2,3,4) = q_2 \end{aligned} \tag{35}$$

We use feed-back FNN and feed-forward FNN shown in Figure 1 and Figure 2 to approximate the solution y. The exact solution is termed as $= (1,3,4)$. The training starts with $y(0) = (3,6,7), \eta = 4 \times 10^{-4}$ as well as $\gamma = 5 \times 10^{-4}$. Table 1 displays the estimated solution considering the number of iterations. The preciseness of the computed solution $y_0(t)$ is portrayed in Figure 4, where t is taken to be the number of iterations.

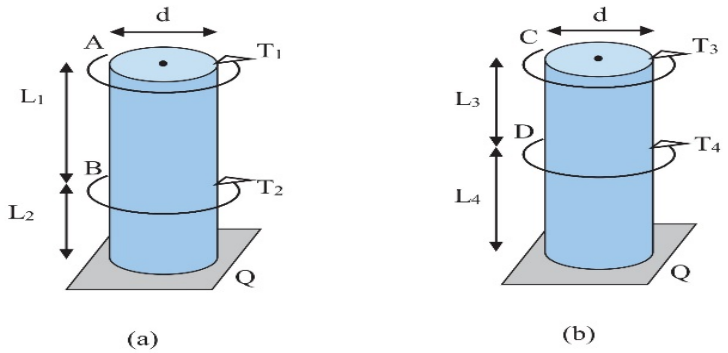

Fig. 3. The cylindrical force

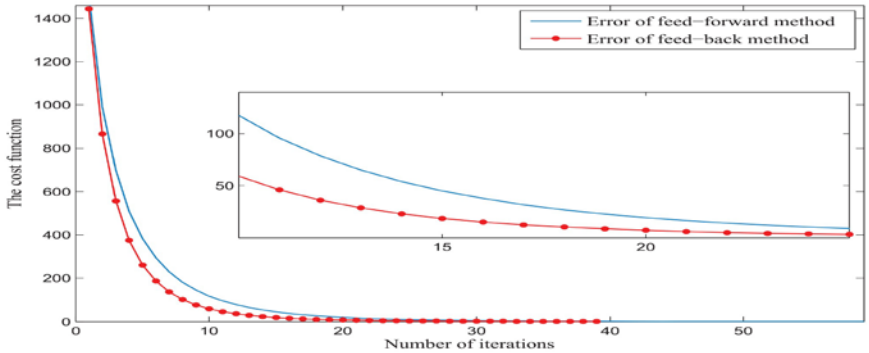

Fig. 4. The cost function associated with Example 2 considering the number of iterations with both techniques

Table 1. The estimated solutions at par with error analysis associated to Example

t	$y_0(t)$ by feed-forward FNN	Error	t	$y_0(t)$ by feed-back FNN	Error
1	(2.9685,5.8452,6.9025)	1443.3563	1	(2.8692,5.7905,6.8479)	1443.3563
2	(2.7245,5.6413,6.6585)	999.52256	2	(2.5892,5.5812,6.4475)	806.67385
3	(2.4998,5.4025,6.3011)	696.92256	3	(2.3098,5.2560,6.0313)	517.06672
4	(2.2458,5.0255,5.9915)	509.34256	4	(2.0514,4.9047,5.7065)	375.04018
5	(2.0150,4.8961,5.6162)	382.84254	5	(1.8256,4.5973,5.3560)	231.27915
⋮	⋮	⋮	⋮	⋮	⋮
55	(1.0099,3.0043,4.0067)	0.32515382	35	(1.0075,3.0081,4.0097)	0.6460055
56	(1.0081,3.0036,4.0052)	0.22514552	36	(1.0068,3.0062,4.0078)	0.4542216
57	(1.0075,3.0029,4.0041)	0.11254587	37	(1.0052,3.0044,4.0057)	0.3298205
58	(1.0061,3.0020,4.0030)	0.12552141	38	(1.0043,3.0029,4.0040)	0.1649085
59	(1.0048,3.0011,4.0022)	0.09254654	39	(1.0035,3.0019,4.0022)	0.0888326

4 Concluding remarks

This paper describes the design and training of a FNN which is used for solving FFDP. To obtain a solution of a FFDP, the adjustable parameter of FNN is systematically adjusted by using a learning algorithm that is based on the gradient descent method. The effectiveness of the derived learning algorithm is demonstrated by computer simulation on numerical examples. We proposed two examples based on applications. The comparison of the feed-back FNN method with the feed-forward FNN method shows that the feed-back FNN method is better or at least more suitable than the feed-forward FNN method. The reason behind it is that, the speed of convergence is increased which depends on the number of computations.

References

1. Jin, L., Gupta, M.M.: Stable dynamic backpropagation learning in recurrent neural networks. IEEE Trans. Neural Networks. 10(6), 1321-1334 (1999).
2. Jafari, R., Yu, W., Li, X.: Numerical solution of fuzzy equations with Z-numbers using neural networks. In: Intelligent automation and Soft Computing, 1-7 (2017).
3. Jafari, R., Yu, W., Li, X., Razvarz, S.: Numerical Solution of Fuzzy Differential Equations with Z-numbers Using Bernstein Neural Networks. In: International Journal of Computational Intelligence Systems, 10(1), 1226–1237(2017).
4. Hussian, E.A., Suhhiem, M.H.: Numerical solution of fuzzy partial differential Equations by using modified fuzzy neural networks. British Journal of Mathematics and Computer Science, 12(2), 1-20 (2016).
5. Jafari, R., Yu, W.: Fuzzy Differential Equation for Nonlinear System Modeling with Bernstein Neural Networks. IEEE Access. doi:10.1109/ACCESS.2017.2647920 (2017).
6. Jafari, R., Yu, W.: Uncertainty Nonlinear Systems Modeling with Fuzzy Equations. Mathematical problems in Engineering. 2017, https://doi .org/10.1155/2017/8594738 (2017).
7. Jafarian, A., Jafari, R., Mohamed Al Qurashi. M., Baleanud. D.: A novel computational approach to approximate fuzzy interpolation polynomials. Springer Plus. 5, 14-28 (2016).

8. Ishibuchi, H., Kwon, K., Tanaka, H.: A learning of fuzzy neural networks with triangular fuzzy weights. Fuzzy Sets and Syst. 71(3), 277-293 (1995).
9. Abbasbandy, S., Otadi, M.: Numerical solution of fuzzy polynomials by fuzzy neural network. Appl. Math. Comput. 181(2), 1084-1089 (2006).
10. Jafarian, A., Measoomynia, S.: Solving fuzzy polynomials using neural nets with a new learning algorithm, Aust. J. Basic and Appl. Sci. 5(9), 2295-2301 (2011).
11. Friedman, M., Ming, Ma., Kandel, A.: Fuzzy linear systems. Fuzzy Sets and Systems. 96(1), 201-209 (1998).
12. Friedman, M., Ming, Ma., Kandel, A.: Duality in fuzzy linear systems. Fuzzy Sets and Systems. 109(1), 55-58 (2000).
13. Otadi, M.: Fully fuzzy polynomial regression with fuzzy neural networks. Neurocomputing, 142, 486-493 (2014).
14. Abbasbandy, S., Asady, B.: Newton's method for solving fuzzy nonlinear equations. Applied Mathematics and Computation. 159(2), 356-379 (2004).
15. Dehghan, M., Hashemi, B.: Iterative solution of fuzzy linear systems. Applied Mathematics and Computation. 175(1), 645-674 (2006).
16. Rouhparvar, H.: Solving fuzzy polynomial equation by ranking method, In: First Joint Congress on Fuzzy and Intelligent Systems. Ferdowsi University of Mashhad, Iran (2007)
17. Nurhakimah Ab. R., Lazim, A.: An Interval Type-2 Dual Fuzzy Polynomial Equations and Ranking Method of Fuzzy Numbers. International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering, 8(1), 92-99 (2014).
18. Muzzioli, S., Reynaerts, H.: The solution of fuzzy linear systems by non-linear programming: a financial application. European Journal of Operational Research. 177(2), 1218-1231 (2007).
19. Amirfakhrian, M.: Numerical solution of algebraic fuzzy equations with crisp variable by Gauss-Newton method. Applied Mathematical Modelling. 32(9), 1859-1868 (2008).
20. Kumar, A., Bansal, A., Babbar, N.: Solution of fully fuzzy linear system with arbitrary coefficients. International Journal of Applied Mathematics and Computation. 3(3), 232-237 (2011).
21. Ezzati, R.: Solving fuzzy linear systems. Soft Computing. 15(1), 193-197 (2011).
22. Allahviranloo, T., Mikaeilvand, N.: Non zero solutions of the fully fuzzy linear systems. Applied and Computational Mathematics. 10(2), 271-282 (2011).
23. Waziri, M.Y., Majid, Z.A.: A new approach for solving dual fuzzy nonlinear equations using Broyden's and Newton's methods. Advances in Fuzzy Systems. 2012(1), 1-5 (2012).
24. Otadi, M., Mosleh, M.: Solution of fuzzy polynomial equations by modified Adomian decomposition method. Soft Computing. 15(1), 187-192 (2011).
25. Allahviranloo, T., Gerami Moazam, L.: The solution of fully fuzzy quadratic equation based on optimization theory. The Scientific World Journal. 2014(1), 1-6 (2014).
26. Babbar, N., Kumar, A., Bansal, A.: Solving fully fuzzy linear system with arbitrary triangular fuzzy numbers (m, α, β). Soft Computing. 17(4), 691-702 (2013).
27. Oh, S.K., Pedrycz, W., Roh, S.B.: Genetically optimized fuzzy polynomial neural networks with fuzzy set-based polynomial neurons. Information Sciences. 176(23), 3490-3519 (2006).
28. Wang, C.C., Tsai, C.F.: Fuzzy processing using polynomial bidirectional heteroassociative network. Information Sciences. 125(1-4), 167-179 (2000).
29. Zadeh, L.A.: Toward a generalized theory of uncertainty (GTU) an outline. Information Sciences. 172(1-2), 1-40 (2005).
30. Beer, F.P. Johnston, E.R.: Mechanics of materials. 2nd Edition, mcgraw-Hill, New York (1992).