Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 6 – No.3, November 2016 – www.caeaccess.org

# Function Approximation using Neural and Fuzzy Methods

Mithaq Nama Raheema
Lecturer, Ph.D. in Electronic Engineering
Electrical Engineering Department, University of Technology
Baghdad, Iraq

Ahmad Shaker Abdullah
PSc. in Electronic Engineering
Electrical Engineering Department, University of Technology
Karbala, Iraq

## ABSTRACT
This work deals with an approximation of functions which finds the underlying relationship from an available finite input-output data of the function. It is the fundamental problem in a majority of real world applications, such as signal processing, prediction, data mining and control system. In this paper five different methods are used to verify their efficiency of approximation: MLPNN, RBFNN, GRNN, FIS and ANFIS networks. The performance is compared by using the RMSE measurement as an indicator of the fitness of these models in function approximation problem. The experimental results show that the performance of all networks used in this work at the training process is more different at the checking process when the networks have been tested with unknown data points. This depends on many factors such as type of networks used to approximate the function, available training data, noise in the data and values of the required parameters for training each network (No. of layers, No. of neurons, No. of training epochs, etc.).

## Keywords
Function Approximation, MLP, GRNN, RBFNN, FIS, ANFIS

## 1. INTRODUCTION
Function approximation is an important task in many different economic, engineering, and computational problems [1], such as pattern recognition, data mining, system identification and control, classification and forecasting [2-4]. From a finite data set, the basic task of a function approximation method is to find the suitable relationship between variables and their corresponding responses [4]. There are different approaches of the function approximation including analytical methods such as least squares linear approximation, polynomial approximation, and shape-preserving approximation in addition to many intelligent methods such as approximation with Fuzzy [5-6], Neural Networks (NNs) [7-8] or combined between neural and fuzzy [3, 9-10]. Both NNs and fuzzy logic can be recommended as universal function approximators, provided that sufficient hidden neurons in NN or rules in fuzzy logic [3] can give good performance for nonlinear function approximation. In this paper, MLPNN, GRNN, RBFNN, FIS and ANFIS networks are applied to approximate five different functions. As an indicator of the accuracy for these five methods the performance is compared by using the root mean square error (RMSE) function.

## 2. METHODS
The five methods used are:

### 2.1 Multilayer Perceptron Neural Network (MLPNN)
MLPNN is an information processing network simulates the biological nervous systems process as in human brain [11]. It involves many layers of nodes. Each layer is linked to the next layer in the network. Input layer receives the input data. Other layers work to map inputs to outputs by performing construction of the inputs with the node's weights by applying activation function. Logistic and hyperbolic tangent sigmoid functions are the most common activation function in MLPNN [12].

### 2.2 Generalized Regression Neural Network (GRNN)
GRNN is a universal approximator that with enough data can approximate many smooth functions by estimating a probability distribution function. GRNN consists of four layers: the input layer and output layer in addition to pattern layer and summation layer [7].

### 2.3 Radial Basis Function Neural Network (RBFNN)
RBFNN is a function approximation model that can map a desired input output relationship once it trained by examples. RBNN, under certain conditions, is capable of approximating arbitrarily well many functions [13]. The performance of the RBFNN depends on the number, centers and shapes of the RBFNNs, and the learning method used for mapping the input output data [13]. RBFNN network involves one hidden layer and a linear output.

### 2.4 Fuzzy Inference System (FIS)
FIS is constitute four different parts: fuzzifier, fuzzy inference engine, fuzzy rule base and defuzzifier. It consists of fuzzy sets represented by membership functions. With Gaussian membership function the fuzzy basis function is can be a universal approximator for many real continuous function [14]. A fuzzy system can approximate a function through covering the graph of function with fuzzy patches and by averaging patches that overlap. The approximation improves as the fuzzy patches grow in number and shrink in size [6]. Since the shapes of the fuzzy sets can affect the accuracy of the approximation, fuzzy sets with varying shapes may be better than those with fixed shapes [15].

### 2.5 Adaptive Neuro-Fuzzy Inference System (ANFIS)
ANFIS is a hybrid artificial intelligent technique infers knowledge by using the principle of fuzzy logic and adds the

possibility of the learning of Artificial Neural Network (ANN). It can be seen as six interconnected layers of ANN, where each layer is equivalent in operation to the output found in a particle stage of a fuzzy system. [16]. Fuzzy logic rules and membership functions can be generated using the ANN learned knowledge which can significantly reduce the development time [17].

This paper focused on using ANFIS in function approximation and compared with other methods.

## 3. SIMULATION

Five different methods of approximation methods are applied to learn the landscape of five separate functions. Firstly, the input and output data are generated using the mathematical formulas of these functions. The total number of data used are 1200 points, 70% of the data are used in training process and other data are used for test. Checking process is done by using all data points of function. Results are presented both numerically and graphically. Simulation is done by using Matlab version 2014a. To evaluate the approximation results the RMSE is chosen as an error criterion for comparison between methods, that is:

$$\mathrm{RMSE} = \sqrt{\sum_{i=1}^{N} \left(y(i) - \left(f(i)\right)^2 / N\right.} \qquad \dots (1)$$

Where f(i) and y(i) are the calculated and exact function values at point i respectively, and N is the total number of points.

MLP network consists of one neuron in the input layer, 12 neurons for the first hidden layer and 6 neurons in the second hidden layer and one neuron for output layer, number of training epochs equal to 200, the activation function is sigmoid function in hidden layers and linear function in output layer. Figure 1 shows the designed MLPNN network. GRNN consists of two hidden layer, the first layer contain 841 neurons, which represent number of training points, with Gaussian activation function, second layer has one neuron of linear function in addition to the input layer and the output layer each of one neuron as shown in Figure 2. The architecture design of RBFNN is similar to GRNN shown in Figure 2 except the first hidden layer has 40 neurons. The architecture design of FIS and ANFIS each consist of one input, one output and Takagi-Sugeno method of fuzzy inference with 3 rules for FIS and 5 rules for ANFIS as shown in Figure 3 and Figure 4.

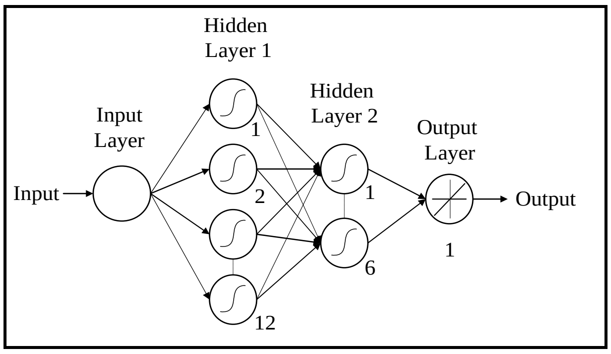

**Fig 1: Network Design of MLPNN**

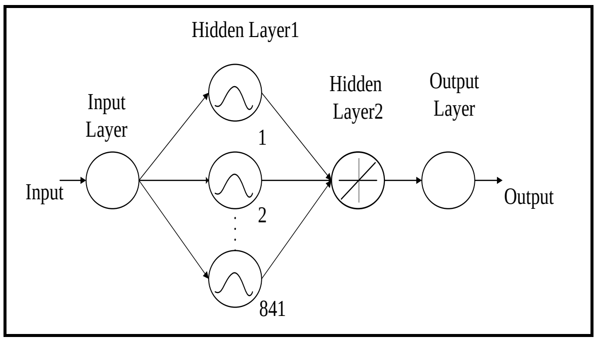

**Fig 2: Network design of GRNN**

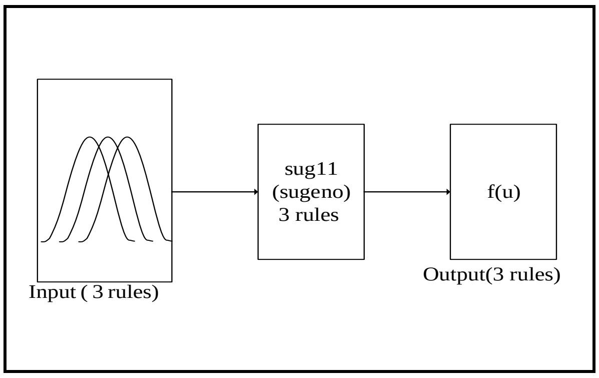

**Fig 3: System diagram of FIS**

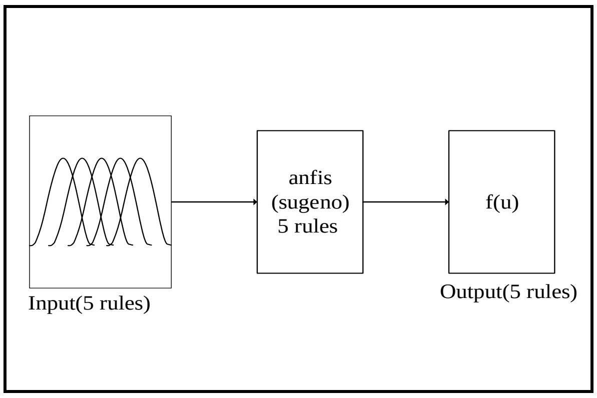

**Fig 4: System diagram of ANFIS**

## 4. RESULTS

In this work samples of five different functions as shown in Table 1 are taken and applied to five methods to be approximated.

**Table 1: Functions used in Simulation Work**

| Case | Functions Name | Function Equation | Range of Data |
|---|---|---|---|
| 1 | Bessel | $f(x) = \lvert \mathrm{besselj}(ax) * \sin^{-1}(bx) + x^c + d \rvert$ | 0 – 10 |
| 2 | Composed of Continuous Exponential Sections | $f(x) = e^{(-5a(x-b)^2} - 5ce^{(-10a(x-b)^2} + \tan(dx) + 2ce^{(-a(x+b)^2} - 2ce^{(-2a(x+1.5b)^2} + ce^{(-a(x+2b)^2} - 2ce^{(-a(x+3b)^2}$ | -6 – 6 |
| 3 | Exponential | $f(x)) = (x+a) * e^{(-bx+c)}$ | -1 – 1 |

| | | | |
|---|---|---|---|
| 4 | Sine and Exponential | $f(x) = \sin(a\pi x) * e^{-\lvert bx \rvert}$ | -1 – 1 |
| 5 | Another form of Sine and Exponential | $f(x) = \frac{\sin(ax)}{e^{\frac{x}{b}}}$ | 0 – 10 |

Where a, b, c, and d are coefficients of equation. Figures 5-9 show the original functions and the available data for functions used in this work.

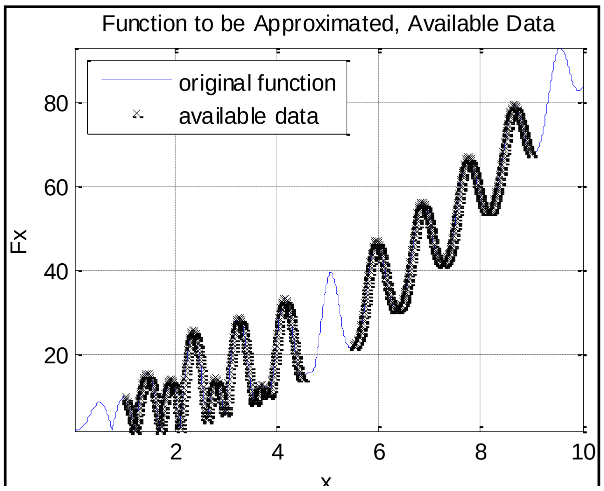

**Fig 5: Function used in case 1**

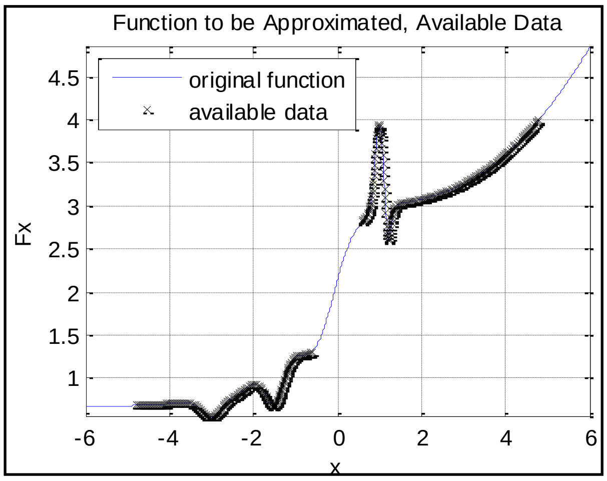

**Fig 6: Function used in case 2**

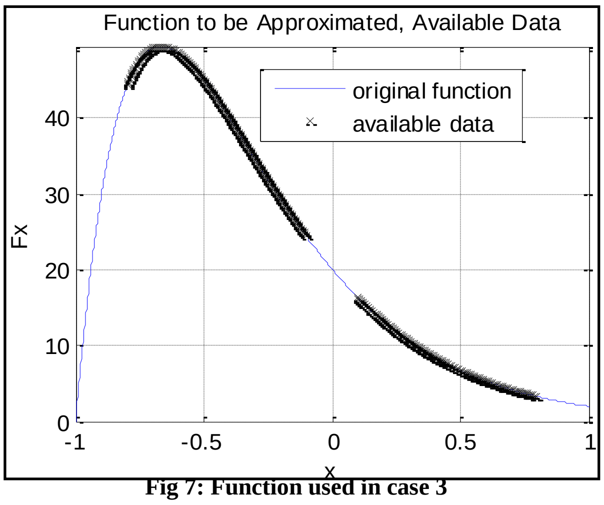

**Fig 7: Function used in case 3**

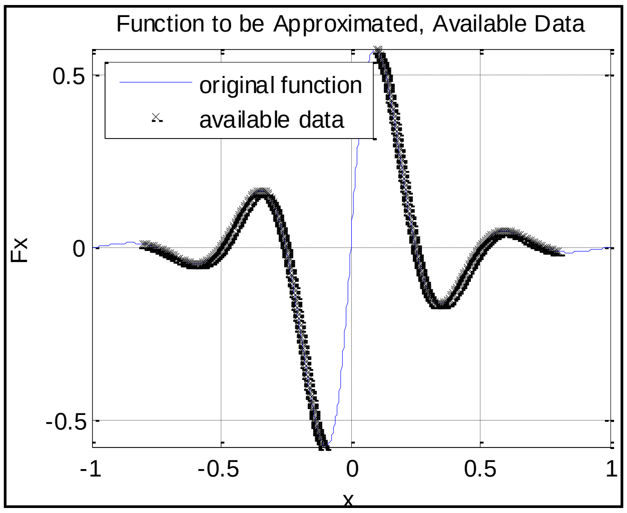

**Fig 8: Function used in case 4**

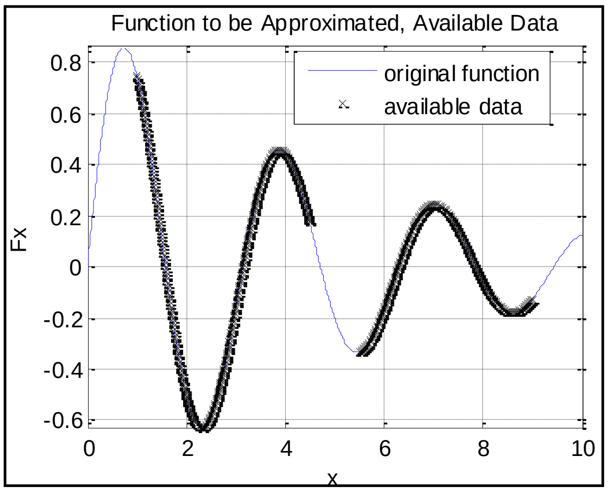

**Fig 9: Function used in case 5**

The simulation results for these case by using MLPNN, GRNN, RBFNN, FIS and ANFIS networks are shown in Figures 10-14. Tables 2-3 show the RMSE for the approximation of these functions in training and checking processing.

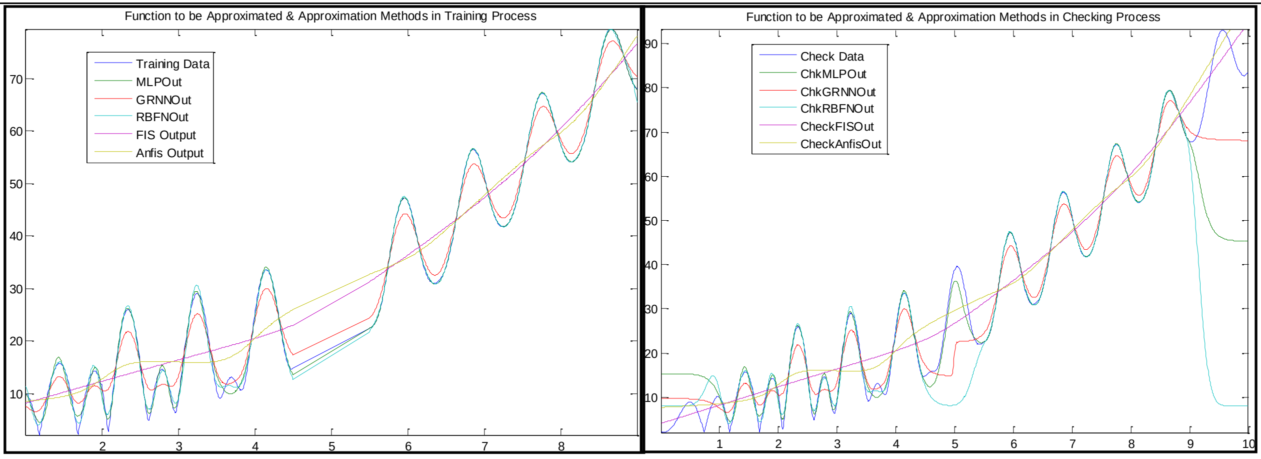

**Fig 10: Comparison between the five methods applied on the function in case 1**

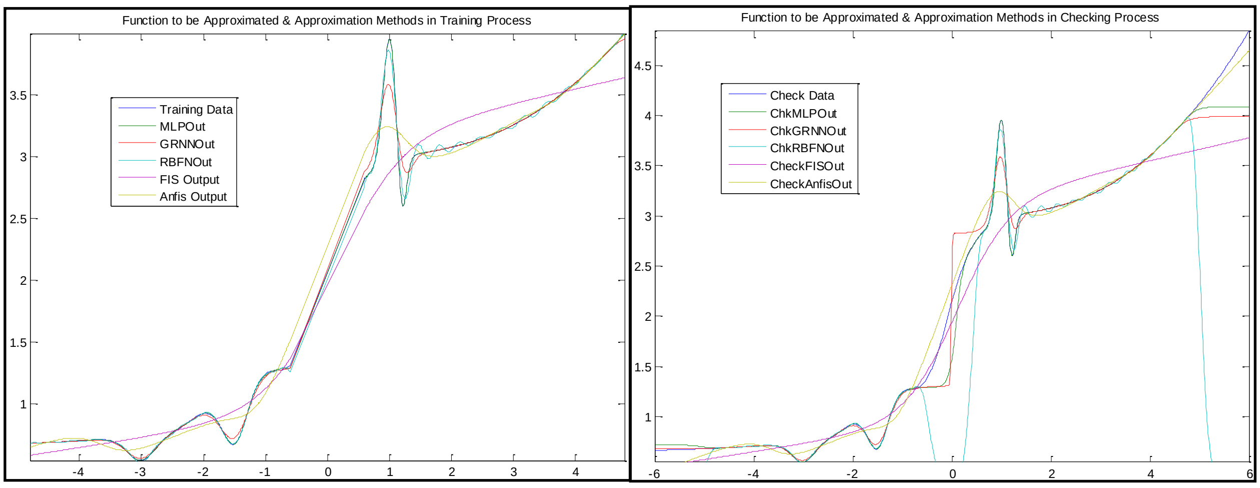

**Fig 11: Comparison between the five methods applied in the function in case 2**

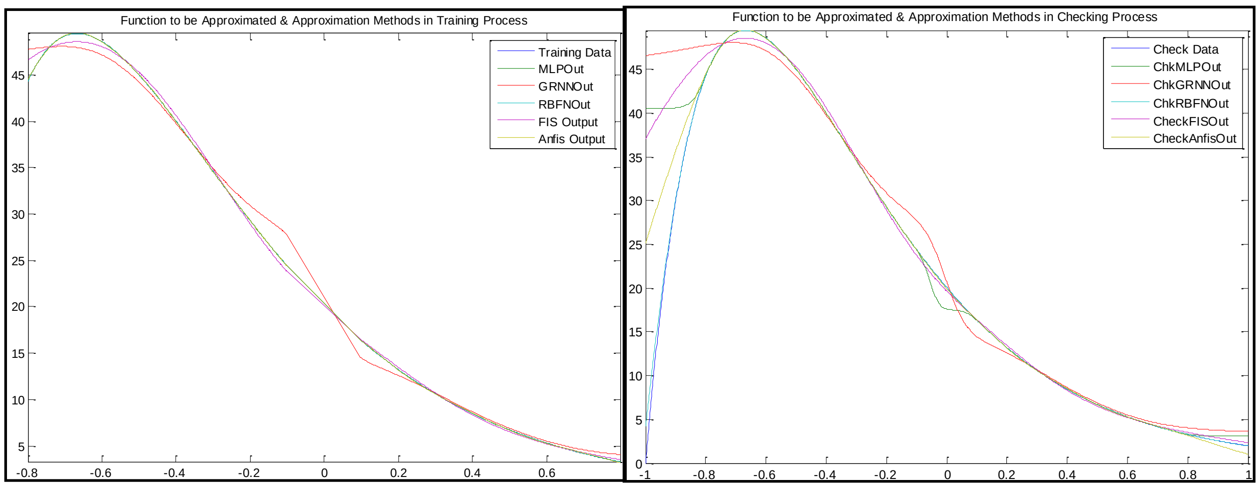

**Fig 12: Comparison between the five methods applied in the function in case 3**

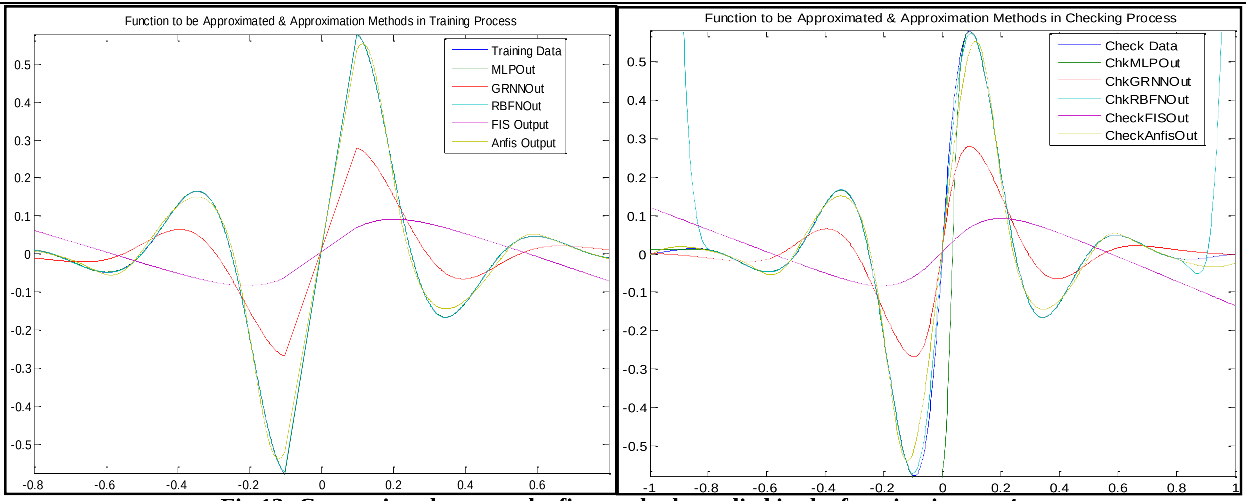

**Fig 13: Comparison between the five methods applied in the function in case 4**

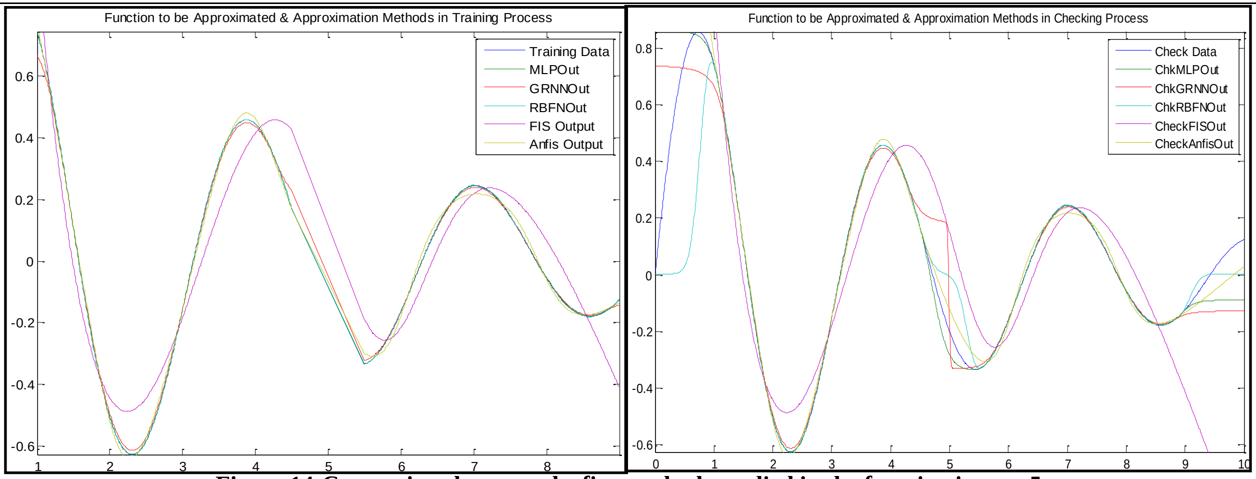

**Figure 14:Comparison between the five methods applied in the function in case 5**

**Table 2: RMSE of training process**

| Method / Function | MLPNN | GRNN | RBFNN | FIS | ANFIS |
|---|---|---|---|---|---|
| Case1 | 0.5617 | 2.3250 | 0.73109 | 6.7889 | 6.7254 |
| Case2 | 0.0025 | 0.0598 | 0.02731 | 0.2088 | 0.1295 |
| Case3 | 0.0091 | 1.0748 | 1.7e-05 | 0.4378 | 0.0447 |
| Case4 | 7.5e-05 | 0.0996 | 0.00078 | 0.1776 | 0.0122 |
| Case5 | 0.0002 | 0.0111 | 0.00141 | 0.1016 | 0.0153 |

**Table (3.3): RMSE of checking process**

| Method / Function | MLPNN | GRNN | RBFNN | FIS | ANFIS |
|---|---|---|---|---|---|
| Case1 | 9.43438 | 6.6357 | 22.4714 | 6.6257 | 6.9448 |
| Case2 | 0.24888 | 0.1964 | 1.22170 | 0.2984 | 0.1389 |
| Case3 | 6.15581 | 7.5431 | 0.41170 | 5.7681 | 3.4487 |
| Case4 | 0.10212 | 0.1105 | 0.97205 | 0.1980 | 0.0361 |
| Case5 | 0.12130 | 0.11672 | 0.160975 | 0.84970 | 0.29880 |

Figure 10 (left) and Tables 2-3 show that the MLPNN and RBFNN networks give good performance for training data of case 1 compared with other network; while MLPNN and RBFNN give large values of RMSE in checking process because of the unknown data. Figure 10 (right) and Tables 2-3 show that the FIS and GRNN networks give lower RMSE values and they can approximate the target function better than other methods in checking data with all training and unknown data.

Results of Figure 11 (left) and the Table 2 present that the MLPNN network gives the best performance of approximation for training data; while from the right sides of Figure 11 and Table 3 the ANFIS network gives best approximation for checking data. The simulation result for case 2 by using ANFIS is shown in Figure 20 compared to the result of MLPNN.

Figure 12 (left) and Tables 2-3 show that the MLPNN, RBFNN and ANFIS networks give small values of RMSE that mean a good performance for training data; while only RBNN network can approximate the function better than other four methods in checking data with all training and unknown data. Figure 21 shows the simulation result for case 3 by using RBFNN. Results of Figure 13 (left) and Tables 2-3 present that the RBFNN network gives the best performance of approximation for training data; and the ANFIS network in checking process. The simulation result for case 4 by using ANFIS is shown in figure 22.

In case 5, it is clear from Figure 14 and Tables 2-3 that the GRNN gives the smallest RMSE in checking process compared with MLPNN and RBFNN networks in the training process. The simulation result for case 5 by using GRNN is shown in figure 23.

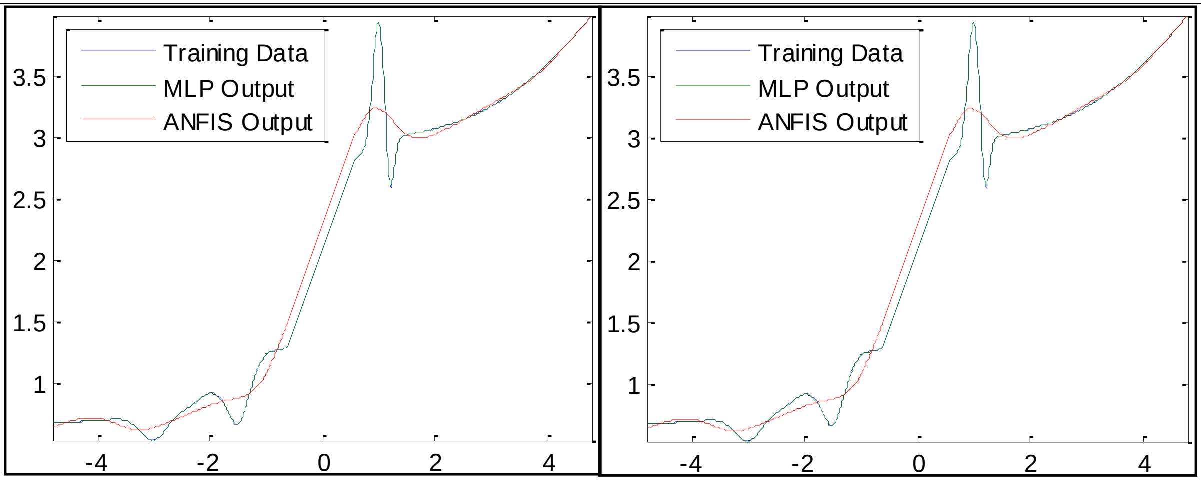

**Fig 20: Simulation result for case 2 using ANFIS (left) training data (right) checking** data

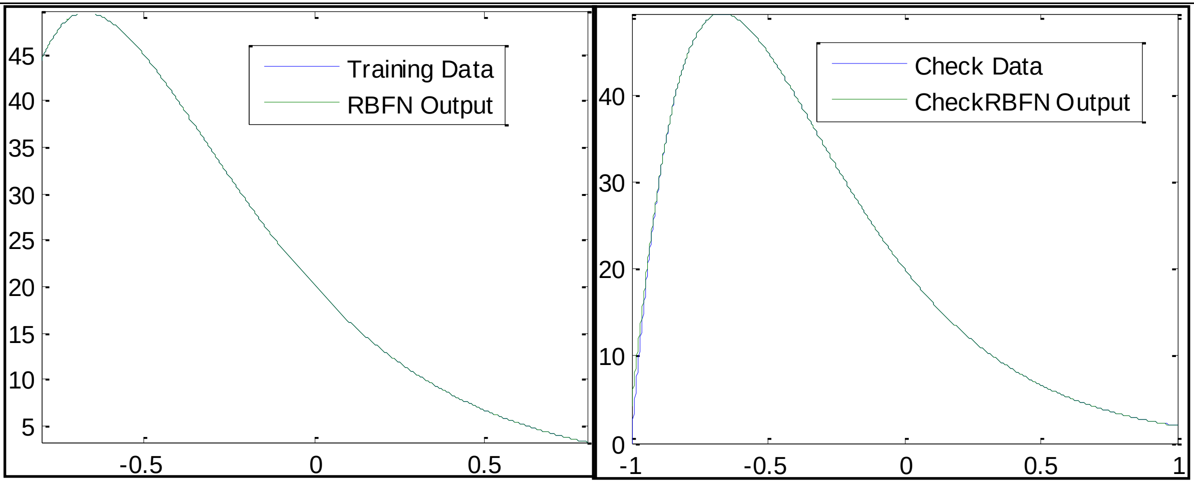

**Fig 21: Simulation result for case 3 using RBFNN (left) training data (right) checking data**

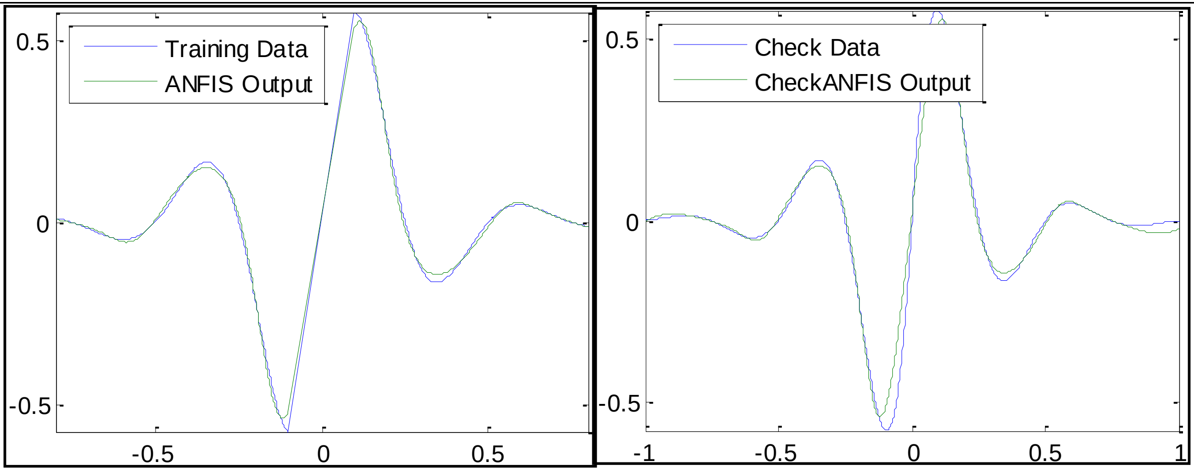

**Fig 22: Simulation result for case 4 using ANFIS (left) training data (right) checking data**

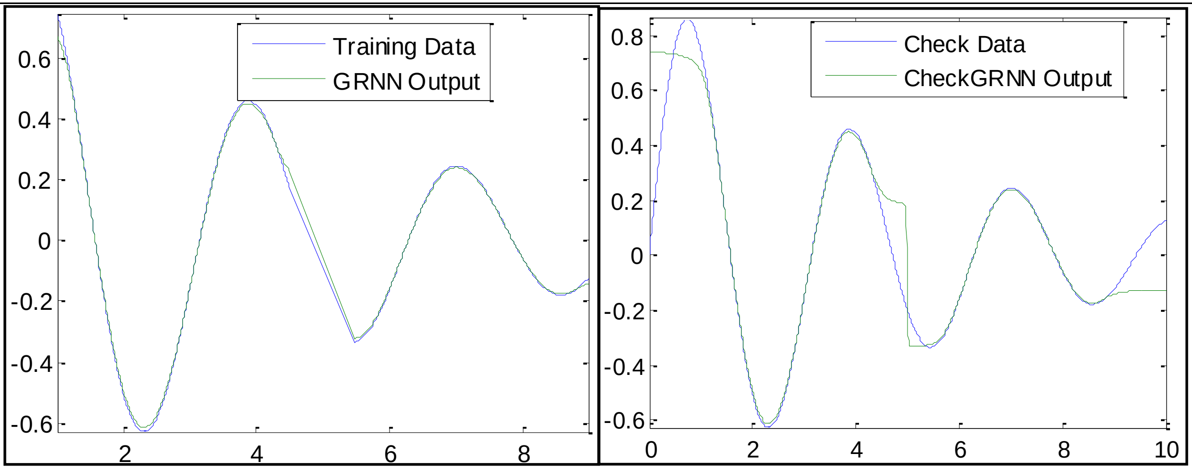

**Figure 23: Simulation result for case 5 using GRNN (left) training data (right) checking data**

Table 4 shows the best training and checking methods for all simulation cases.

**Table 4: Best method for each cases**

| Case | Best Training Method | Best Checking Method |
|---|---|---|
| 1 | MLPNN | FIS |
| 2 | MLPNN | ANFIS |
| 3 | MLPNN | RBFNN |
| 4 | RBFNN | ANFIS |
| 5 | MLPNN | GRNN |

As shown from table 4 results are divided into two part, first part is the training results while second part is checking results. The training results show that the MLP networks provide the best approximation in four cases and RBNN for one case. The checking result show that the best network to approximation function has been distributed into different networks as GRNN, RBNN, FIS, or ANFIS.

## 5. CONCLUSION

This work focus on checking process more than training process in choice of the best network to approximate any function because of the checking process use the data presented in training process in addition to the unknown data that didn’t enter in the training process. From Table 4 the GRNN, RBFNN, and FIS models yield good results each in one case; the ANFIS approach yields a superior result in two cases. From the simulation results, one can conclude that there is no specific method to approximate all functions, it is depends on the type of the function to be approximate and the method that used. In this paper, the ANFIS has been used for the first time in function approximation problem and it proves its ability in this application. Any of networks can be considered as a good approximator if one changes factors entered in training process such as number of neurons, type of activation or member function, or increase the number of training epochs etc. Triangle, trapezoid, and Gaussian membership functions are the most common fuzzy sets use in FIS and ANFIS networks, one may compare these functions to study their different effects in function approximation problem.

## 6. REFERENCES

[1] Peter Andras, "Function Approximation Using Combined Unsupervised and Supervised Learning," IEEE, vol. 25, March 2014.

[2] Zarita Zainuddin, Ong Pauline, "Function approximation using artificial neural networks," International Journal of Systems Applications, Engineering & Development vol. 1, no. 4, 2007.

[3] Horng-Lin Shieh, Ying-Kuei Yang, Po-Lun Chang, Jin-Tsong Jeng, "Robust neural-fuzzy method for function approximation," Elsevier Ltd, p. 6903–6913, 2008.

[4] YunfengWu, "Adaptive Linear and Normalized Combination of Radial Basis Function Networks for Function Approximation and Regression," springer, 2014.

[5] Zarita Zainuddin, Pauline Ong, "Design of wavelet neural networks based on symmetry fuzzy C-means for function approximation," springer, 29 January 2013.

[6] Julie A. Dickerson, Bart Kosko, "Fuzzy Function Approximation with Ellipsoidal Rules," IEEE, vol. 26, pp. 542-560, August 1996.

[7] Salim Heddam, Abdelmalek Bermad, Noureddine Dechemi, "Applications of Radial-Basis Function and Generalized Regression Neural Networks for Modeling of Coagulant Dosage in a Drinking Water-Treatment Plant: Comparative Study," Environment Engineering, December 2011.

[8] Mohammed Salem, Meriem Amina Zingla, Mohamed Faycal Khelfi, "An Enhanced Swarm Intelligence based Training Algorithm for RBF Neural Networks in Function Approximation," IEEE, 2014.

[9] Omid Khayat, Javad Razjouyan , Fereidoun Nowshiravan Rahatabad,Hadi Chahkandi Nejad, "A fast learnt fuzzy neural network for huge scale discrete data function approximationand prediction," Intelligent and fuzzy systems, May 2013.

[10] Andre Lemos, Vladik Kreinovich, Walmir Caminhas, Fernando Gomide, "Universal Approximation with Uninorm-Based Fuzzy Neural Networks," IEEE, 2011.

[11] Ramandeep Kaur, Prince Verma, "Improved MLP-NN based approach for Lung Diseases Classification," International Journal of Computer Applications, vol. 131, December2015.

[12] Gurpreet Singh, Manoj Sachan, "Multi-Layer Perceptron (MLP) Neural Network Technique for Offline Handwritten Gurmukhi Character Recognition," IEEE International Conference on Computational Intelligence and Computing Research, 2014.

[13] Nicolaos Karayiannis, "On the Construction and Training of Reformulated Radial Basis Function Neural Networks," IEEE, vol. 14, pp. 835-846, JULY 2003.

[14] Kah Wai Cheah, Noor Atinah Ahmad, "Universal Approximation of Reduced Fuzzy Basis Function With Ruspini Partitioning," ResearchGate, October 2015.

[15] Hung-Ching Lu, Sing-Fu Lee, "Function Approximation of Nonlinear Functions by GA-Based Fuzzy Systems," IEEE, July 2012.

[16] Jose Kleton, Nonlinear system idenification based on modified ANFIS, 2003.

[17] Lakhmi C. Jain; N.M. Martin, Fusion of Neural Networks, Fuzzy Systems and Genetic Algorithms: Industrial Applications, 1998.