A NEW FUZZY WAVELET NEURAL NETWORK DESIGN FOR TIME SERIES PREDICTION

Sevcan YILMAZ
Master of Science Thesis

Computer Engineering Program
June, 2009

JÜRİ VE ENSTİTÜ ONAYI

Sevcan Yılmaz'ın **"A New Fuzzy Wavelet Neural Network Design for Time Series Prediction"** başlıklı **Bilgisayar Mühendisliği** Anabilim Dalındaki, Yüksek Lisans Tezi 12.06.2009 tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

	Adı-Soyadı	**İmza**
Üye (Tez Danışmanı)	**: Doç. Dr. YUSUF OYSAL**	**………………..**
Üye	**: Prof. Dr. YAŞAR HOŞCAN**	**………………..**
Üye	**: Doç. Dr. AYDIN AYBAR**	**………………..**

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ……………… tarih ve ………… sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

ABSTRACT

Master of Science Thesis

A NEW FUZZY WAVELET NEURAL NETWORK DESIGN FOR TIME SERIES PREDICTION

Sevcan YILMAZ

Anadolu University
Graduate School of Sciences
Computer Engineering Program

Supervisor: Assoc. Prof. Dr. Yusuf OYSAL

2009, 58 pages

In this thesis, two different neuro-fuzzy models which use wavelet basis functions in its processing units are proposed for time series prediction and system identification problems. The structure of introduced models comes from the idea of adaptive neuro-fuzzy inference system (ANFIS) which is used for obtaining fuzzy rule base from the input-output data of an unknown function. The first model in this thesis is called as adaptive wavelet network (AWN) in which wavelet basis functions are used as membership functions in antecedent part of the rules whereas mostly Gaussian type membership functions are used in the ANFIS. In the second model which is called as fuzzy wavelet neural network (FWNN), wavelet basis functions are used in consequent part of the rules instead of zero or first order polynomial function in the ANFIS. A fast training gradient algorithm based on quasi-Newton methods is used to obtain the optimal values for unknown parameters of the FWNN models. The AWN models are trained by a hybrid algorithm which combines gradient algorithm with least square estimation. Simulation examples of some benchmark problems in the literature are also given to illustrate the effectiveness of models.

Keywords: Fuzzy Wavelet Neural Networks, System Identification, Time Series Prediction, Wavelet, Wavelet Neural Networks

ÖZET

Yüksek Lisans Tezi

ZAMAN SERİSİ KESTİRİMİ İÇİN
YENİ BİR DALGACIK BULANIK AĞ TASARIMI

Sevcan YILMAZ

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Yusuf OYSAL
2009, 58 sayfa

Bu tezde, işlem birimlerinde dalgacık fonksiyonu kullanan iki farklı bulanık sinir ağı modeli zaman serisi kestirimi ve sistem tanımlaması problemleri için önerilmiştir. Bu modellerin yapısı, bilinmeyen bir fonksiyonun giriş-çıkış verisinden bulanık kural tabanını elde etmek için kullanılan adaptif sinirsel bulanık çıkarım sistemi (ASBÇS) fikrinden gelmektedir. Bu tezde adaptif dalgacık ağ (ADA) olarak adlandırılan birinci modelde dalgacık fonksiyonları kuralların koşul kısmında üyelik fonksiyonu olarak kullanılmaktadır. ASBÇS'de ise genellikle Gaussian tipindeki üyelik fonksiyonları kullanılmaktadır. Bulanık dalgacık sinir ağı (BDSA) olarak adlandırılan ikinci modelde, dalgacık fonksiyonları kuralların sonuç kısımlarında ASBÇS' deki sıfırıncı ya da birinci dereceden polinom yerine kullanılmıştır. Yaklaşık Newton yöntemine dayanan hızlı bir gradyan eğitim algoritması BDSA modellerinin bilinmeyen parametrelerinin optimal değerlerini bulmak için kullanılmıştır. ADA modelleri gradyan algoritmasını en küçük kareler yöntemiyle birleştiren bir hibrit algoritma kullanılarak eğitilmiştir. Literatürde ölçüt olarak kullanılan bazı benzetim örnekleri de modellerin etkisini göstermek için verilmiştir.

Anahtar Kelimeler: Bulanık Dalgacık Sinir Ağları, Sistem Tanıması, Zaman Serisi Kestirimi, Dalgacık, Dalgacık Sinir Ağı

ACKNOWLEDGEMENTS

I would like to thank my supervisor Assoc. Prof. Dr. Yusuf Oysal for his guidance and support during my study. It was my pleasure to work with him during this study. Also, I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) and TURKCELL Academy for giving me financial support through their scholarship programmes during this study. Finally, I would like to express my special thanks to my family for their moral and financial support all the time.

Sevcan YILMAZ

June, 2009

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ANFIS	: Adaptive Neuro-Fuzzy Inference System
AWN	: Adaptive Wavelet Network
AWN-F	: First Order Adaptive Wavelet Network
AWN-Z	: Zero Order Adaptive Wavelet Network
BFGS	: Broyden-Fletcher-Goldfarb-Shanno
FWNN	: Fuzzy Wavelet Neural Network
FWNN-M	: Multiplication Fuzzy Wavelet Neural Network
FWNN-R	: Radial Fuzzy Wavelet Neural Network
FWNN-S	: Summation Fuzzy Wavelet Neural Network
LSE	: Least Square Estimation
MF	: Membership Function
MSE	: Mean Square Error
NMSE	: Normalized Mean Square Error
NN	: Neural Network
RBF	: Radial Basis Function
RMSE	: Root Mean Square Error
SISO	: Single Input Single Output
WNN	: Wavelet Neural Network

1. INTRODUCTION

In recent years, wavelets have become very popular and have been applied in many scientific and engineering research areas such as system identification, signal processing, function approximation, pattern recognition and data preprocessing. The idea of combining wavelets with neural networks (NN) has led to the development of wavelet neural networks (WNN). WNNs can be easily trained and give high accuracy because of time-frequency localization properties of wavelets.

On the other hand, neuro-fuzzy systems which integrate learning capability of neural networks with perfect inference mechanism of fuzzy systems are also used for nonlinear function approximation, system identification and pattern recognition. These neuro-fuzzy systems have fast and accurate learning, good generalization capabilities and the ability to accommodate both data and expert knowledge about the problem under consideration.

In this thesis, the aim is to combine neuro-fuzzy systems with wavelet functions to increase the performance of neuro-fuzzy systems for system identification and time series prediction problems.

The rest of this chapter is organized as follows. The detailed information about NNs will be given in Section 1.1. While wavelets will be introduced in Section 1.2, WNNs will be illustrated in Section 1.3. In Section 1.4, the properties of neuro-fuzzy systems will be described. Fuzzy wavelet neural networks (FWNN) will be explained in Section 1.5. Finally, organization of thesis will be presented in Section 1.6.

1.1. Neural Networks

NNs are nonlinear mapping structures based on the function of human brain. They are powerful tools for modeling, especially when the underlying data relationship is unknown. A NN involves a number of processing units called neurons which communicate by sending signals to each other over a large number of weighted connections [1]. Neurons can carry out their computations in parallel. NNs derive their computing power through two important characteristics [2]. These are parallel distributed structure of NNs and generalization property of

NNs. Generalization refers to the NN producing reasonable outputs to the inputs not encountered during training (learning). These two information-processing capabilities make it possible for NNs to solve complex problems.

There are several types of NNs. Some of them are feed-forward NNs, recurrent NNs and radial basis function (RBF) networks. In feed-forward NN, the data flow from input to output is strictly feed-forward and there can be multiple layers of neurons but no feedback connections are present[1]. Recurrent NNs contain feedback connections and the dynamical properties of recurrent NN are important contrary to feed-forward NNs [1]. RBF networks use RBFs as activation functions. RBF networks typically have three layers: an input layer, a hidden layer with radial basis activation functions and output layer. RBF networks are universal approximators. This means that a RBF network with enough hidden neurons can approximate any multivariate continuous function with arbitrary precision [2].

1.2. Wavelets

Wavelets are functions that satisfy certain mathematical requirements and are used in representing data or other functions. They are generated by one function $\Psi(x)$, called the mother wavelet, by dilation and translation:

$$\psi_{b,c}(x) = \frac{1}{\sqrt{c}}\psi\left(\frac{x-b}{c}\right) \tag{1.1}$$

In (1.1), *c* is the dilation factor and *b* is the translation factor and the factor $c^{-1/2}$ is for energy normalization across the different scales.

The mother wavelet satisfies the following condition called compatibility condition:

$$\int_0^\infty \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < \infty \tag{1.2}$$

where $\Psi(\omega)$ is the Fourier transform of $\Psi(x)$. The compatibility condition implies the average value of the wavelet in the time domain must be zero:

$$\int_{-\infty}^{\infty} \Psi(x)dx = 0 \tag{1.3}$$

and therefore wavelets must be oscillatory, in other words $\Psi(x)$ must be a wave.

A certain class of functions can be represented as linear combinations of the wavelets. That is, the functions are represented as finite linear combinations of the translated and dilated versions of wavelet functions. Wavelet representations are more efficient than Fourier series representations where the signal changes its behavior with time [3, 4]. Both wavelet and Fourier representations are localized in frequency however wavelets are localized in time while Fourier sine and cosine functions are not [5]. This feature makes wavelets more efficient than Fourier representations in function approximation. In addition, a wavelet is more flexible since we can choose suitable mother wavelet from various types of wavelets according to our signal to be analyzed whereas Fourier representation has fixed basis namely the sine and cosine functions [4].

1.3. Wavelet Neural Networks (WNN)

The idea of combining both wavelets and NNs has resulted in the formulation of WNNs [6-8]. NNs have learning and generalization abilities, nonlinear mapping and parallelism of computation however they require large number of neurons for approximation problems. In addition, their convergence is generally slow. However, WNNs converge quickly, can be easily trained and give high accuracy because of time-frequency localization properties of wavelets. The main characteristic of a WNN is that wavelet functions are used in hidden layer of NN as activation functions instead of local functions in time such as Gaussian and sigmoid functions. The WNN is a nonlinear regression structure which represents input-output mappings by dilated and translated versions of wavelet functions which have time-frequency localization properties.

At present, there are two kinds of WNN structure. In the first one, wavelets as activation functions stem from continuous wavelet transform. Therefore, dilation and translation parameters of wavelet function can be any real positive number and these parameters and output layer weights can be adjustable. In the second type, again wavelets as activation functions stem from discrete wavelet transform. But in this case the dilation and translation parameters of wavelet functions are fixed, and only the output layer weights are adjustable.

Several WNN models are proposed in literature [9-12]. In [9], a local linear wavelet neural network is presented and it is an example of first type of WNNs mentioned above. In this network, connection weights between hidden layer and output layer are replaced with a local linear model whereas in conventional WNNs, these weights are adjustable constant values. In [10], a linear wavelet network which combines conventional WNN with weighted linear summation of inputs is proposed. In [11], the WNN which is second type of wavelet networks mentioned above is proposed for medium and high dimensional problems, it decomposes a multidimensional function into a number of low-dimensional sub-models which are expanded using wavelet decomposition. In [12], Bayesian approach is applied to wavelet networks for nonparametric regression.

1.4. Neuro-Fuzzy Systems

Every intelligent technique such as NNs, fuzzy logic has particular computational properties and these make them suitable for particular problems. While NNs offer advantages such as learning, parallelism and generalization, they are not explaining how they reach their decisions [13]. Fuzzy logic systems are good at explaining their decisions but they cannot automatically acquire the rules they use to make those decisions [13]. Therefore, combination of NNs and fuzzy logic systems which is called neuro-fuzzy system overcomes the limitations of individual techniques. Neuro-fuzzy systems can obtain the fuzzy rule base from the given input-output data.

One of the popular neuro-fuzzy systems is adaptive neuro-fuzzy inference system (ANFIS) [14]. ANFIS is functionally equivalent to Sugeno’s inference mechanism. In a Sugeno fuzzy model, input space is divided into fuzzy regions and these regions show fuzzy membership functions in the antecedent part of the fuzzy rules. The consequents of these rules are represented by either a constant or a linear function of inputs. The structure of the ANFIS is explained in detail in Chapter 2.

1.5. Fuzzy Wavelet Neural Networks

FWNNs combine wavelets with neuro-fuzzy systems in order to increase the power of these techniques significantly. FWNNs take advantages and properties of NNs, fuzzy logic and wavelets. These are generalization and learning ability of NNs, time-frequency localization properties of wavelets and inference mechanism of fuzzy logic systems.

In the literature, several FWNN models are proposed for time series prediction, system identification and control problems [15-22]. In [15] and [16], each fuzzy rule is represented by a sub-WNN which consists of single-scaling wavelets that is same dilation parameter for all dimensions and orthogonal least-square algorithm is used to select important wavelets. The resulting network is used for function approximation in [15] and control of nonlinear systems in [16]. In [17], the proposed model consists of a set of if–then rules and, then parts are series expansion in terms of wavelets functions and this model is applied to system modeling. In [18], both sigmoid and wavelet functions are used in hidden layer of WNN and the output of this new WNN is calculated by multiplication and summation of these results. Then, this WNN is used in consequent parts of if-then rules in FWNN. In [19], a dynamic recurrent fuzzy wavelet network for identified nonlinear dynamic systems is proposed. In [20], the inputs enter into discrete wavelet transform block, the output of this block is fuzzified and it forms the input to a single neural network and this model is used for system identification and control problems. The FWNN proposed in [21] uses summation of dilated and translated versions of wavelet functions in consequent part of fuzzy rules for system identification and control purposes.

1.6. Organization of the Thesis

The organization of this thesis is as follows. In Chapter 2, the structure and learning algorithm of ANFIS will be explained in detail and then the work in this thesis will be introduced. In Chapter 3 and Chapter 4, two different neuro-fuzzy models proposed in this thesis which use wavelet functions in their processing units are explained in detail with their structures and learning algorithms. These models are tested for both time series prediction and system identification problems and simulation results for these tests are given in Chapter 5. Finally, conclusion is drawn in Chapter 6.

2. BACKGROUND

In this chapter, firstly structure and learning algorithm of the ANFIS will be explained in detail in sections 2.1 and 2.2 respectively. Then, least squares estimation (LSE) will be described in Section 2.3 and finally, the aim and the proposed work in this thesis will be introduced in Section 2.4.

2.1 Adaptive Neuro-Fuzzy Inference System

The ANFIS is a Sugeno type fuzzy system that put in the framework of adaptive systems to facilitate learning and adaptation [14]. In a Sugeno fuzzy system, input space is divided into fuzzy regions and these regions show fuzzy membership functions in the antecedent part of the fuzzy rules. These membership functions can be triangular, bell shape or Gaussian type functions. The consequents of these rules are represented by either a constant or a linear function of inputs. The ANFIS structure is organized to give the output of this predefined fuzzy system. Unknown parameters of the fuzzy rules are learnt by hybrid learning algorithm.

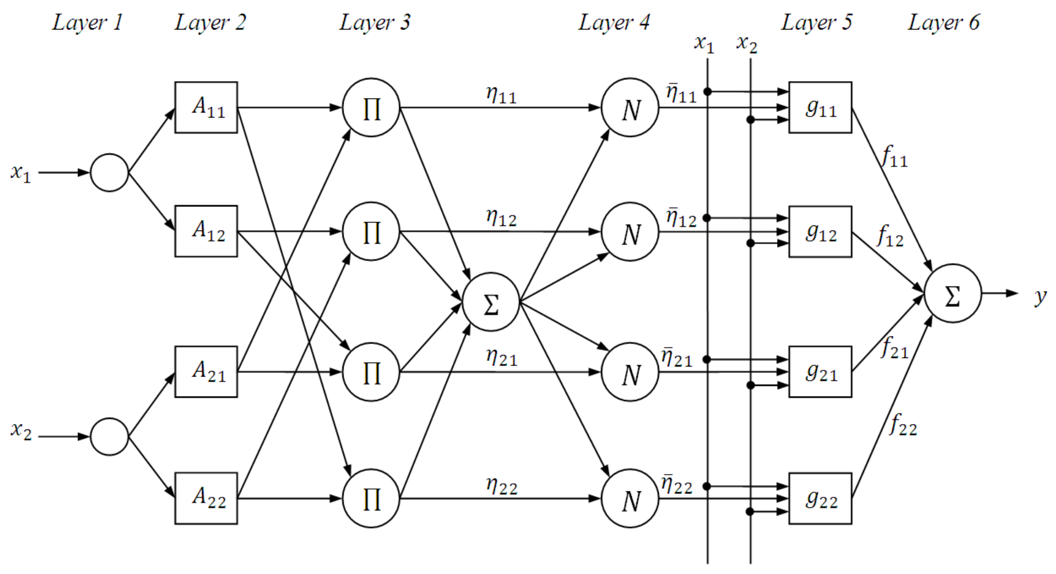

Figure 2.1. The structure of ANFIS

For simplicity, the ANFIS to be explained here has two inputs and one output, and each input has two membership functions and output is the linear combination of first order polynomial of input variables. For example, the first fuzzy rule of this ANFIS is defined as follows:

IF x_1 is A_{11} **AND** x_2 is A_{21} **THEN** $g_{11} = p_{11}x_1 + q_{11}x_2 + r_{11}$

The ANFIS structure is shown in Figure 2.1. The node functions in the same layer are selected from the same function family as described below:

- ***Layer 1:*** This layer is the input layer. Each neuron in this layer transmits external crisp input signals (x_1 and x_2) directly to the next layer.
- ***Layer 2:*** This layer is fuzzification layer. Neurons in this layer represent fuzzy sets used in the antecedents of fuzzy rules. The outputs of this layer are the values of the membership functions. The *j*th Gaussian type membership function for the *i*th input is given by:

$$A_{ij}(x_i) = \exp(-\frac{1}{2}(\frac{x_i - \mu_{ij}}{\sigma_{ij}})^2) \quad i=1,2 \text{ and } j=1,2 \tag{2.1}$$

- ***Layer 3:*** This layer is fuzzy rule layer. Each neuron in this layer represents activation strength of a fuzzy rule. The output of this layer is equal to multiplication of coming signals.

$$\eta_{ij} = A_{1i}(x_1) \cdot A_{2j}(x_2) \qquad i=1,2 \text{ and } j=1,2 \tag{2.2}$$

- ***Layer 4:*** This layer is normalization layer. Each neuron in this layer calculates the normalized activation strength of a given rule by:

$$\overline{\eta}_{ij} = \frac{\eta_{ij}}{\sum_{i=1}^{2}\sum_{j=1}^{2}\eta_{ij}} \qquad i=1,2 \text{ and } j=1,2 \tag{2.3}$$

- ***Layer 5:*** This layer is defuzzification layer. Each neuron in this layer receives initial inputs (x_1 and x_2) and the normalized activation strengths calculated previous layer as input and calculates the weighted consequent value of a given combination as follows:

$$\begin{aligned} f_{ij} &= \overline{\eta}_{ij} g_{ij} \\ &= \overline{\eta}_{ij}(p_{ij}x_1 + q_{ij}x_2 + r_{ij}) \end{aligned} \tag{2.4}$$

- ***Layer 6:*** This layer is output layer. It computes the overall output of system as follows:

$$y = \sum_{i=1}^{2}\sum_{j=1}^{2} f_{ij} \qquad i=1,2 \text{ and } j=1,2 \tag{2.5}$$

2.2 Hybrid Learning Algorithm for the ANFIS

The unknown parameters of the ANFIS networks are nonlinear parameters (**μ** and **σ**) in premise part of the rules and linear parameters (***p***, ***q*** and ***r***) in consequent part of the rules. The combination of gradient method and least squares estimation (LSE) method is used to update the unknown parameters of the ANFIS networks. Each epoch of this hybrid learning procedure consist of two stages: a forward pass and a backward pass. In the forward pass, outputs of all layers up to 6th layer are calculated and consequent parameters are updated using LSE method. In the backward pass, error functional in (2.6) is calculated and premise parameters are updated using gradient descent algorithm. Error functional is defined as:

$$E = \frac{1}{2}\sum_{k=1}^{N} e^2 \tag{2.6}$$

where e shows the prediction error. In gradient descent algorithm, gradients of the error functional with respect to all unknown parameters are required. Parameter update rules for nonlinear parameters are defined by following formulas:

$$\mu_{k+1} = \mu_k - \tau \frac{\partial E}{\partial \mu_k} \tag{2.7}$$

$$\sigma_{k+1} = \sigma_k - \tau \frac{\partial E}{\partial \sigma_k} \tag{2.8}$$

where τ is step size in the algorithm. The error rate of overall output is reduced each iteration of the algorithm. The stages of hybrid algorithm are summarized in Table 2.1.

Table 2.1. Two stages in the hybrid learning algorithm for the ANFIS

	Forward Pass	**Backward Pass**
Premise Parameters	Fixed	Gradient Descent
Consequent Parameters	LSE	Fixed

2.3 Least Square Estimation

In the general least squares problem, the output of a linear model y is given by linearly parameterized expression [23]:

$$y = f_1(u)\theta_1 + f_2(u)\theta_2 + \ldots + f_n(u)\theta_n \tag{2.9}$$

where $\boldsymbol{u}$ is the model's input vector, f_1, ⋯ , f_n are known functions of $\boldsymbol{u}$, and θ_1,⋯,θ_n are unknown parameters to be estimated. If we have a training set with m input-output pairs, our system is defined as follows:

$$\begin{cases} f_1(u_1)\theta_1 + f_2(u_1)\theta_2 + \ldots + f_n(u_1)\theta_n = y_1 \\ f_1(u_2)\theta_1 + f_2(u_2)\theta_2 + \ldots + f_n(u_2)\theta_n = y_2 \\ \vdots \\ f_1(u_m)\theta_1 + f_2(u_m)\theta_2 + \ldots + f_n(u_m)\theta_n = y_m \end{cases} \tag{2.10}$$

where $m \geq n$ needs to be solved in parameters θ_i with minimal error. LSE method solves this problem with matrix arithmetic as;
$A\theta = y$, such that;

$$A = \begin{bmatrix} f_1(u_1) & \cdots & f_n(u_1) \\ \vdots & \ddots & \vdots \\ f_1(u_1) & \cdots & f_n(u_m) \end{bmatrix}_{m\times n} , \quad \theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}_{n\times 1} , \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}_{m\times 1} \tag{2.11}$$

For conditions where $m > n$, an exact solution cannot be found since the matrix θ is non-square. To help the solution, an error parameter is added to (2.10) and a performance criteria is obtained:

$$E(\theta) = \frac{1}{2}e^T e = \frac{1}{2}(y - A\theta)^T (y - A\theta) \tag{2.12}$$

and we obtain a prediction value of parameters $\hat{\theta}$ which will minimize (2.12) as:

$$\hat{\theta} = \left(A^T A\right)^{-1} A^T y \tag{2.13}$$

When training the ANFIS, A is a matrix consisting of normalized activation strength ($\overline{\eta}$), x_1 and x_2 values, and θ is a vector whose element values are p, q and r.

2.4 The Proposed Networks in this Thesis

In this thesis, two different neuro-fuzzy systems which use wavelet functions in their processing units are designed for system identification and time series prediction problems. As explained in first chapter, wavelet functions have important properties such as time-frequency localization and these properties make wavelets suitable for function approximation problems. The structures of proposed networks are similar with ANFIS networks. The differences between ANFIS and proposed models are shown in Figure 2.2. The first network which is called as adaptive wavelet network (AWN) in this thesis uses wavelets as membership functions in the second layer of the ANFIS structure whereas the ANFIS networks use generally Gaussian type membership functions. Both constant function and first order polynomial of inputs are used in the consequent part of the rules and this is same with ANFIS networks. The AWN model will be explained in Chapter 3. The second network which is called as fuzzy wavelet network (FWNN) uses wavelets in the consequent part of the rules instead of constant function or first order polynomial of inputs in the ANFIS. The premise part of the rules is same with ANFIS networks. The FWNN model will be explained in Chapter 4.

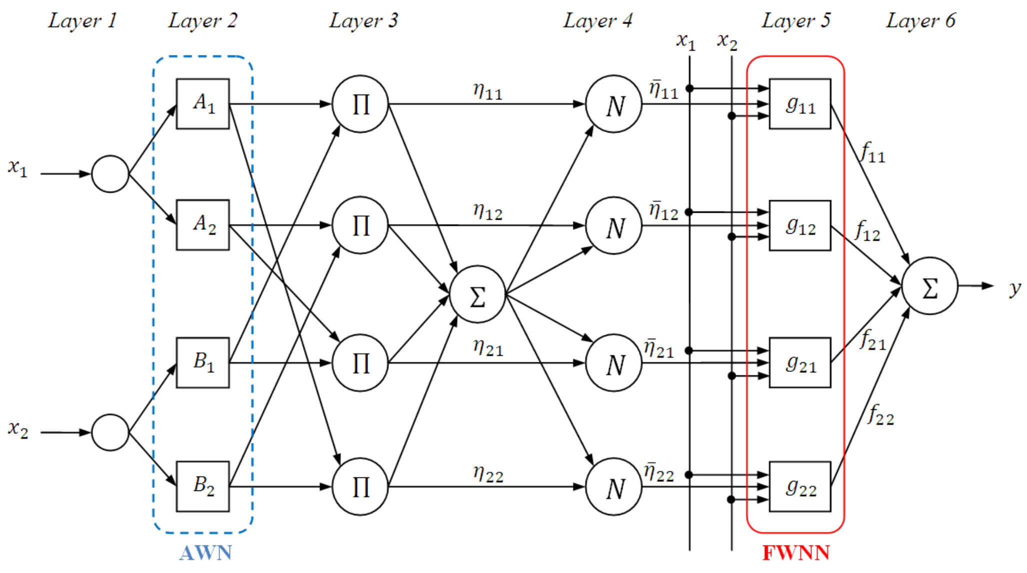

Figure 2.2. The difference between ANFIS and AWN (dashed line), and between ANFIS and FWNN (solid line)

3. ADAPTIVE WAVELET NETWORK (AWN)

In this chapter, AWN will be described. The structure of AWN is similar to the ANFIS. This network uses wavelet basis functions in second layer as membership functions while ANFIS uses generally Gaussian type membership functions. A hybrid learning algorithm which combines LSE and Broyden-Fletcher-Goldfarb-Shanno (BFGS) gradient method is used for training the network.

3.1. Gaussian and Wavelet Basis Functions

In the ANFIS, membership functions are local basis functions such as Gaussian function shown in Figure 3.1. The local basis functions are active for only certain inputs. However, Gaussian function is not local in frequency as seen in Figure 3.2. The locality features both in time and frequency is a very important concept for representation of the signals. Therefore, the mission of the wavelet functions is comprehensive.

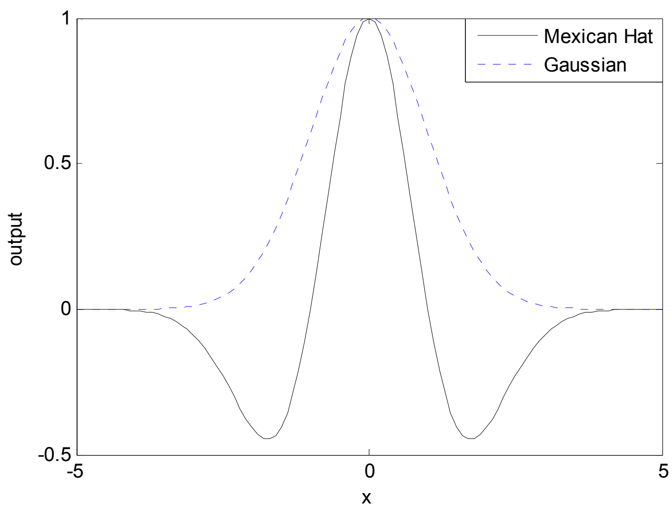

Figure 3.1. Gaussian and Mexican Hat functions

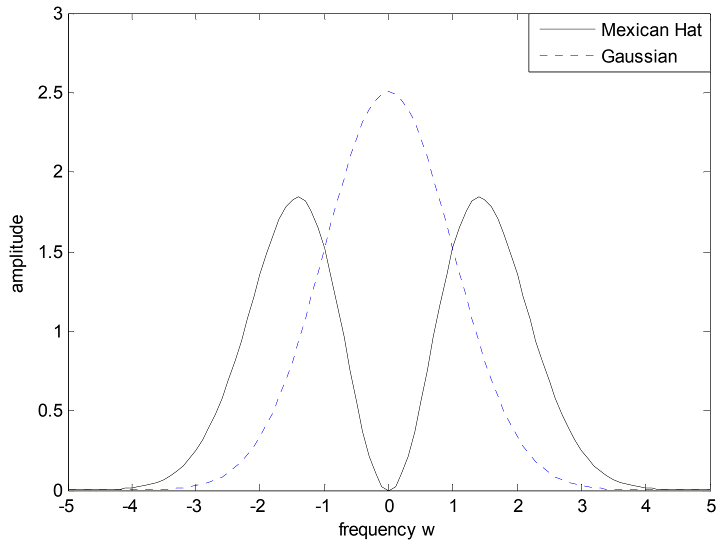

Figure 3.2. The Fourier transform of Gaussian and Mexican Hat wavelet functions

Mexican Hat wavelet function is the second derivative of Gaussian function. Non-orthonormal Mexican Hat basis function can be easily written in the analytical form and its Fourier transform can be easily calculated as given below:

$$\psi(x) = (1 - x^2)\exp(-\frac{x^2}{2}),\ x \in R \tag{3.1}$$

$$\psi(\omega) = \sqrt{2\pi}\omega^2 \exp(-\frac{\omega^2}{2}),\ \omega \in R \tag{3.2}$$

where ω is a real frequency. The Mexican Hat wavelet function can be generalized with translation (b) and dilation (c) parameters as follows:

$$\psi(\frac{x-b}{c}) = (1 - (\frac{x-b}{c})^2)\exp(-\frac{1}{2}(\frac{x-b}{c})^2) \tag{3.3}$$

Translation parameter determines the center position of the wavelet as shown in Figure 3.3, whereas dilation parameter controls the spread of the wavelet as shown in Figure 3.4.

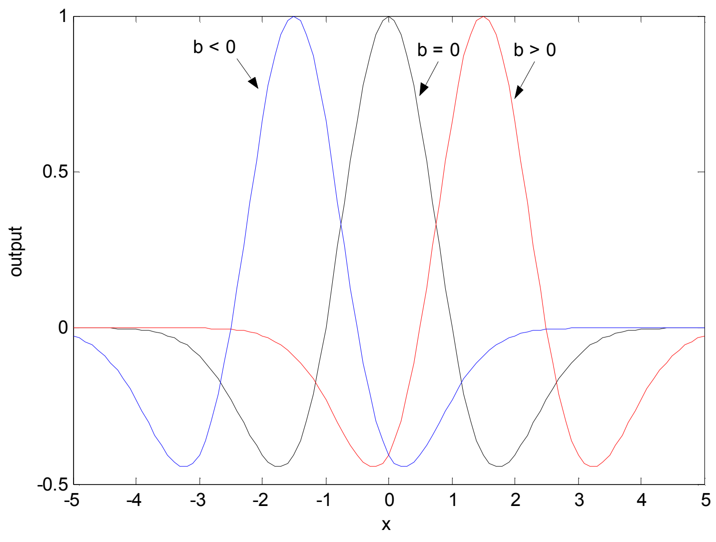

Figure 3.3. Illustration of the translation parameter effect on Mexican Hat wavelet function

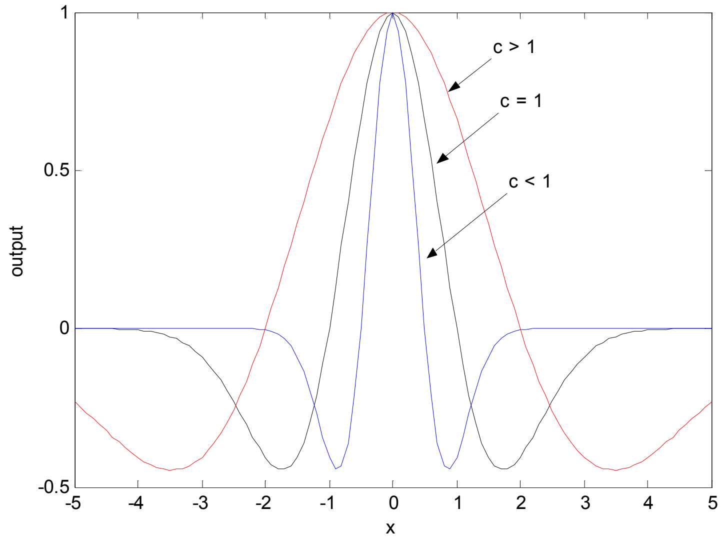

Figure 3.4. Illustration of the dilation parameter effect on Mexican Hat wavelet function

The time and frequency envelope of the Mexican Hat function is shown in Figure 3.4 and Figure 3.5, respectively. Wavelet functions have efficient time-frequency localization properties as shown from the frequency spectrum [24]. If the dilation parameter is changed, the support region width of the wavelet function changes, but the number of cycles doesn't change. That is, the peak number doesn't change. However, when the dilation parameter decreases, the peak point of the spectrum shifts to higher frequency. Therefore, all frequency spectrums can be obtained by changing the dilation parameter.

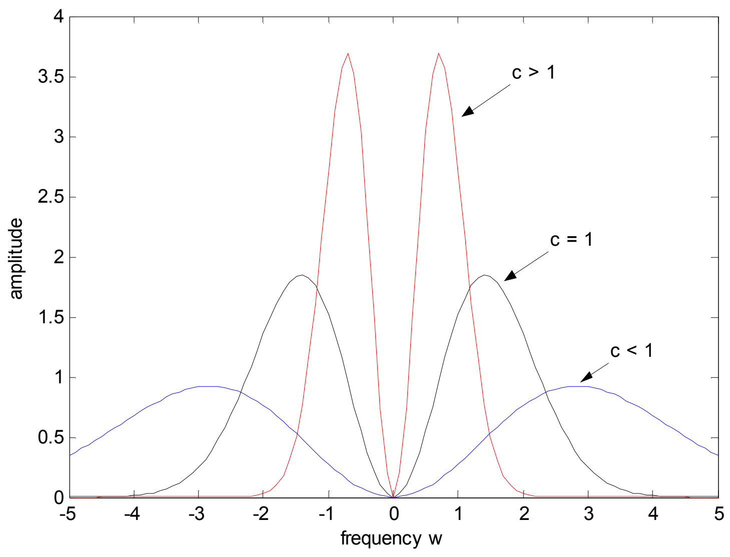

Figure 3.5. Illustration of the dilation parameter effect on Mexican Hat wavelet function's Fourier transform

A deficiency of Gaussian based ANFIS networks doesn't have localization capability in frequency. As shown in Figure 3.2, Gaussian function is not local in frequency. Therefore, it is very difficult to use Gaussian based functions in some applications [25]. To overcome these problems, it is very effective way to use wavelet functions with time-frequency localization properties.

3.2. Structure of the Adaptive Wavelet Network

The new network called AWN uses Mexican hat wavelet function as membership functions. Consequent part of the rules is either constant function or first order polynomial like the ANFIS networks. The six layer structure of the AWN is shown in Figure 3.6.

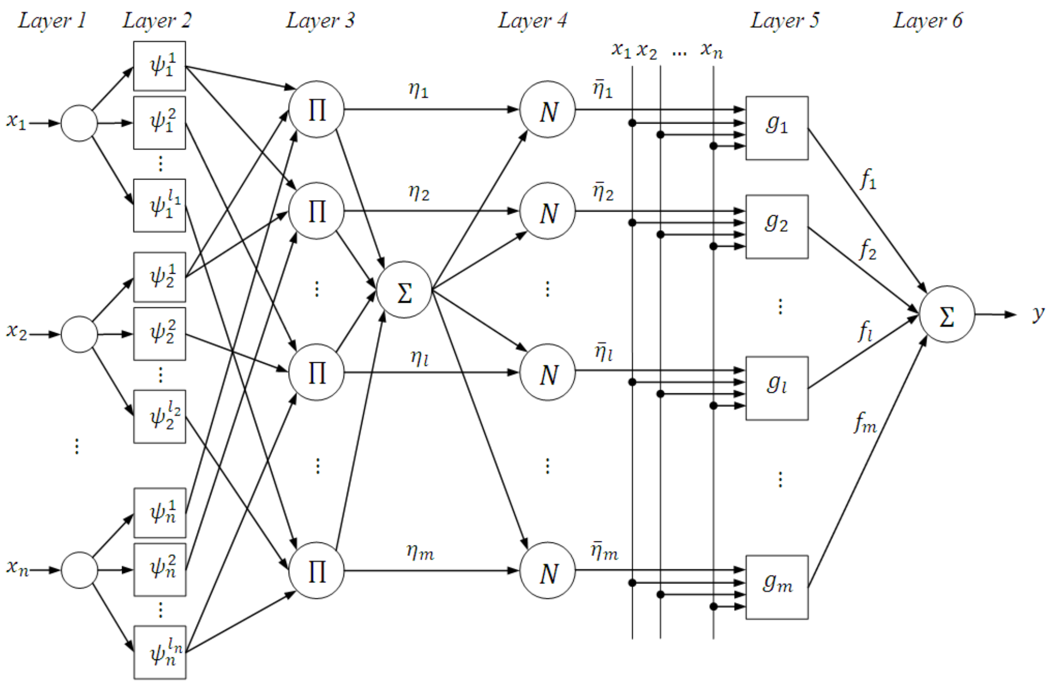

Figure 3.6. The structure of AWN

- ***Layer 1:*** This layer is the input layer. Neurons in this layer simply cross the crisp input signals $x_1, x_2, \dots, x_n$ to the second layer.
- ***Layer 2:*** The outputs of this layer are the values of wavelet basis functions which are used as membership functions. There are total l_1 membership functions for the first input, l_2 for the second input and so on. Following equation shows i_jth membership function for jth input variable.

$$\Psi_j^{i_j} = (1-(\frac{x_j - b_{i_j}}{c_{i_j}})^2)\exp(-\frac{1}{2}(\frac{x_j - b_{i_j}}{c_{i_j}})^2) \tag{3.4}$$

where $j = 1, 2, \dots, n$ and $i_j = 1, 2, \dots, l_j$

- ***Layer 3:*** The activation strength of the each combination of the inputs is represented by the product operator in the AWN network in this layer. Thus, the output of a neuron in the third layer is:

$$\eta_l = \prod_{j=1}^{n} \Psi_j^{i_j}(x_j) \tag{3.5}$$

where $l = i_1 i_2 \dots i_n$, $i_1 = 1, \dots, l_1$, $i_2 = 1, \dots, l_2, \dots, i_n = 1, \dots, l_n$

Each possible combination of membership functions of all inputs represents a rule in this layer. Thus, total number of rules m is given by:

$$m = \prod_{i=1}^{n} l_i \tag{3.6}$$

- ***Layer 4:*** This layer is the normalization layer. Each neuron in this layer calculates the normalized activation strength of a given rule by:

$$\bar{\eta}_l = \frac{\eta_l}{\sum_{i=1}^{m} \eta_i} \qquad (l = 1,...,m) \tag{3.7}$$

- ***Layer 5:*** This layer calculates the weighted consequent value of a given combination as:

$$f_l = \bar{\eta}_l g_l \qquad (l = 1,...,m) \tag{3.8}$$

Two types of AWN model are proposed in this thesis. The first model is zero order AWN (AWN-Z) model where g_l is a constant function. The second model is first order AWN (AWN-F) model where g_l is a first order polynomial function of inputs, i.e.

$$g_l = k_l \qquad \text{for AWN-Z model} \tag{3.9}$$

$$g_l = k_l + \sum_{i=1}^{n} p_{il} x_i \qquad \text{for AWN-F model} \tag{3.10}$$

The first four layers and last layer are same for both zero and first order AWN models. Only the fifth layer differs in these models.

- ***Layer 6:*** This layer contains only a single node and it computes the overall output as the summation of all incoming signals, which is given by:

$$y = \sum_{l=1}^{m} f_l \tag{3.11}$$

3.3. A Hybrid Training Algorithm for AWNs

The total AWN network parameters to be trained are translation parameters (***b***) and dilation parameters (***c***) of wavelet basis functions, and zero and first order model output layer parameters (***k*** and/or ***p***). The number of these unknown parameters is given in Table 3.1 for each AWN model. For example, in two inputs one output model with two wavelet membership functions for each input, there are 8 wavelet function parameters and 4 constant output layer parameters for zero order AWN model. If this network is a first order AWN model, the number of wavelet function parameters is the same; however there will be 12 first order polynomial parameters in the output layer to be determined.

Table 3.1. The number of unknown parameters for the AWN models

Parameters / Models	***b***	***c***	***k***	***p***
Zero order AWN	$\sum_{i=1}^{n} l_i$	$\sum_{i=1}^{n} l_i$	m	0
First order AWN	$\sum_{i=1}^{n} l_i$	$\sum_{i=1}^{n} l_i$	m	nm

The AWN training is to encapsulate a given arbitrary function or input-output pairs by adjusting network parameters. This is done by minimizing the error functional. For the purpose of training an AWN, mean square error (MSE) is selected as a performance index which is given by:

$$E = \frac{1}{N}\sum_{k=1}^{N}(y - y_d)^2 \tag{3.1}$$

N is the number of input-output pairs of the function to be approximated, y_d is the desired output, and y is the AWN output.

Training process is achieved in two stage: The first stage, which is referred to as “forward pass”, consists of calculating the outputs of all layers up to 6th layer by randomly initializing wavelet function parameters (***b*** and ***c***) concerning the input variable ranges and finding the initial values of output layer parameters **(*k*** and/or ***p*)** using LSE. The second part named “backward pass” is completed only when performance index is calculated and, wavelet function and output layer parameters are updated using BFGS gradient method. This hybrid training algorithm for AWN is similar to training algorithm for the ANFIS networks. However, LSE is used only once at the beginning for initialization of output layer parameters. Then, BFGS gradient method has been used for updating of all unknown parameters at each epoch.

3.3.1. Broyden-Fletcher-Goldfarb-Shanno gradient method

BFGS is one of the approximate second order algorithms and derived from Newton's method in optimization which is a class of hill-climbing optimization techniques. It tries to seek the stationary point of a function, where the gradient is zero [26]. It is assumed that at the each iteration of the training algorithm, gradients of the performance index with respect to all unknown parameters (***p***) of the network, $g = \frac{\partial E}{\partial p}$ is computed. Parameter update rules of BFGS algorithm is given by:

$$p^{k+1} = p^k + \tau^k d^k \tag{3.13}$$

$$\tau^k = \min_{\tau} \mathrm{J}(\mathrm{p}^{\mathrm{k}} + \tau d^k) \tag{3.14}$$

$$d^k = -H^k g_p^k \tag{3.15}$$

Here ***p*** is the parameter to be updated, ***d*** is the search direction, ***τ*** is the optimal step size along the search direction, ***g*** is the cost gradient with respect to parameter ***p*** and $H \cong (\nabla_p J)^{-1}$ is the inverse of the approximate Hessian matrix given by:

$$\begin{aligned} H^{k+1} &= \left[I - \frac{\Delta p^k (\Delta g_p^k)^T}{(\Delta p^k)^T \Delta g_p^k} \right] \mathrm{H}^{\mathrm{k}} \left[I - \frac{\Delta p^k (\Delta g_p^k)^T}{(\Delta p^k)^T \Delta g_p^k} \right]^{\mathrm{T}} \\ &+ \frac{\Delta p^k (\Delta p^k)^T}{(\Delta p^k)^T \Delta g_p^k} \ , \ \mathrm{H}^0 = I \end{aligned} \tag{3.16}$$

Δp and Δg are the backward differences of the parameter and gradient vectors, respectively. They provide the history of parameter and gradient changes yielding approximate second order information. The method is faster than the simple gradient method and more robust than simple conjugate gradient approach [26].

3.3.2. Gradients of unknown parameters for the AWN

The cost gradients with respect to network parameters are required for a gradient based algorithm. These are calculated by the following formulas for *i =1…,n*; *l =1,…,m* and *j = 1,…,n* :

$$\frac{\partial E}{\partial k_l}=\frac{\partial E}{\partial y}\overline{\eta}_l \tag{3.17}$$

$$\frac{\partial E}{\partial p_{il}}=\frac{\partial E}{\partial y}\overline{\eta}_l x_i \text{ for AWN-F model} \tag{3.18}$$

$$\frac{\partial E}{\partial b_{i_j}}=\frac{\partial E}{\partial y}\frac{\partial y}{\partial \Psi_j^{i_j}}\frac{\left(x_j-b_{i_j}\right)}{c_{i_j}^2}\left(3-\left(\frac{x_j-b_{i_j}}{c_{i_j}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_j-b_{i_j}}{c_{i_j}}\right)^2\right) \tag{3.19}$$

$$\frac{\partial E}{\partial c_{i_j}}=\frac{\partial E}{\partial y}\frac{\partial y}{\partial \Psi_j^{i_j}}\frac{\left(x_j-b_{i_j}\right)^2}{c_{i_j}^3}\left(3-\left(\frac{x_j-b_{i_j}}{c_{i_j}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_j-b_{i_j}}{c_{i_j}}\right)^2\right) \tag{3.20}$$

$$\frac{\partial E}{\partial y}=\frac{2}{N}\sum_{k=1}^{N}(y-y_d) \tag{3.21}$$

Here for the above calculations, partial derivative of the output *y* with respect to membership functions of each input variable is needed. For the first variable, this can be calculated as follows for *i₁=1,…,l₁*:

$$\frac{\partial y}{\partial \Psi_1^{i_1}(x_1)}=\frac{\sum_{i_2=1}^{l_2}\cdots\sum_{i_n=1}^{l_n} g_{i_1,i_2,\ldots,i_n}\prod_{k=2}^{n}\Psi_k^{i_k}(x_k)-y\sum_{i_2=1}^{l_2}\cdots\sum_{i_n=1}^{l_n}\prod_{k=2}^{n}\Psi_k^{i_k}(x_k)}{\sum_{i_1=1}^{l_1}\sum_{i_2=1}^{l_2}\cdots\sum_{i_n=1}^{l_n}\prod_{k=1}^{n}\Psi_k^{i_k}(x_j)} \tag{3.22}$$

The calculations can be easily generalized for the other input variable membership functions.

For single-input-single-output (SISO) AWN model, the output can be calculated using (3.23) where x_1 is input variable which has l_1 membership functions. The partial derivative of membership functions in (3.22) becomes (3.24) for SISO AWN model.

$$y = \frac{\sum_{i_1=1}^{l_1} g_{i_1} \Psi_1^{i_1}(x_1)}{\sum_{i_1=1}^{l_1} \Psi_1^{i_1}(x_1)} \tag{3.23}$$

$$\frac{\partial y}{\partial \Psi_1^{i_1}(x_1)} = \frac{(g_{i_1} - y)}{\sum_{k_1=1}^{l_1} \Psi_1^{k_1}(x_1)} \qquad (i = 1,\ldots,l_1) \tag{3.24}$$

4. FUZZY WAVELET NEURAL NETWORK (FWNN)

The FWNN models proposed in this thesis are obtained from the traditional Sugeno fuzzy system by replacing the consequent part of fuzzy rules with wavelet basis functions that have time and frequency localization properties. Three type of FWNN models are proposed in this thesis. The first and last model use summation and multiplication of dilated and translated versions of single dimensional wavelet basis functions, respectively and in the second model, consequent parts of the rules consist of radial function of wavelets. A fast training gradient algorithm, BFGS, is used to obtain the optimal values for unknown parameters of the FWNN models.

4.1 The Structure of Wavelet Neural Network

The main characteristic of a WNN is that wavelet functions are used in hidden layer of neural network as activation functions instead of local functions in time such as Gaussian and sigmoid functions. The structure of a WNN shown in Figure 4.1 involves three layers: an input layer, a hidden layer, an output layer. The output of a WNN is given by:

$$y = \sum_{i=1}^{k} w_i \Psi_i(x) = \sum_{i=1}^{k} w_i \left|c_i\right|^{-1/2} \psi\left(\frac{x-b_i}{c_i}\right) \tag{4.1}$$

where ψ_i is the wavelet activation function of *i*th unit of hidden layer and w_i is the weight between hidden and output layer. $\boldsymbol{x}$ is input vector, b_i and c_i are translation and dilation parameters of wavelet functions.

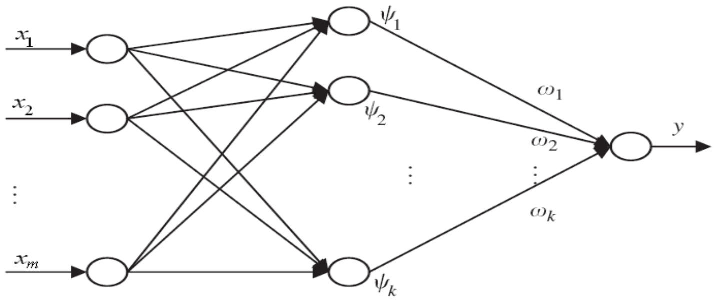

Figure 4.1. The structure of Wavelet Neural Network

Wavelet functions have efficient time-frequency localization properties. Localization of the *i*th unit of hidden layer is determined by translation and dilation parameters of a wavelet function. As explained in previous chapter, translation parameter determines the center position of the wavelet, whereas dilation parameter controls the spread of the wavelet.

When input space is *n* dimensional, wavelet basis function can be calculated by product of *n* single dimensional wavelet basis function.

$$\psi(x) = \psi(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} \psi(x_i) \tag{4.2}$$

Another popular scheme to represent multidimensional wavelets is to choose the wavelets to be some radial functions. For example, *n* dimensional Mexican hat wavelet can be expressed as

$$\psi(x) = (1 - \|x\|^2)\exp(-\frac{\|x\|^2}{2}) \tag{4.3}$$

where $\|x\|^2 = x^T x = \sum_{i=1}^{n} x_i^2$.

4.2. The Structure of Fuzzy Wavelet Neural Network Models

The FWNN models combine Sugeno fuzzy system with wavelet functions. In a Sugeno fuzzy model, input space is divided into fuzzy regions and each region shows a fuzzy membership functions for an input variable. The consequents of fuzzy rules are represented by either a constant or a linear function of inputs. In this part of the thesis, constant or linear functions in consequent part of the rules are substituted with wavelet functions in order to increase computational power of neuro-fuzzy systems because, wavelets have multiresolution property. This property is very useful for function approximation problems. The wavelets can capture global (low frequency) and local (high frequency) behavior of any function easily. This characteristic leads to the proposed FWNN to be of the advantages of fast convergence, easy training and high accuracy. The rules are in following form:

IF x_1 is $A_1^{i_1}$ **AND** x_2 is $A_2^{i_2}$ **AND** … **AND** x_n is $A_n^{i_n}$ **THEN** Ψ_l (x) where x_1, $x_2,\ldots,x_n$ are input variables, $A_j^{i_j}$ is i_jth Gaussian type membership function (MF) for *j*th input and Ψ_l (x) which is a function of inputs is *l*th output of fuzzy rule. In this study, three different FWNN models are proposed according to consequent part of the rules. The first one is summation FWNN (FWNN-S) which uses summation of translated and dilated versions of single dimensional wavelet functions in consequent part of the rules. Multiplication of single dimensional wavelet functions are used in the second model which is called as multiplication FWNN (FWNN-M). In the last model, radial FWNN (FWNN-R), consequent part of the rules consists of radial function of wavelets.

The six layer structure of the FWNN models are shown in Figure 4.2.

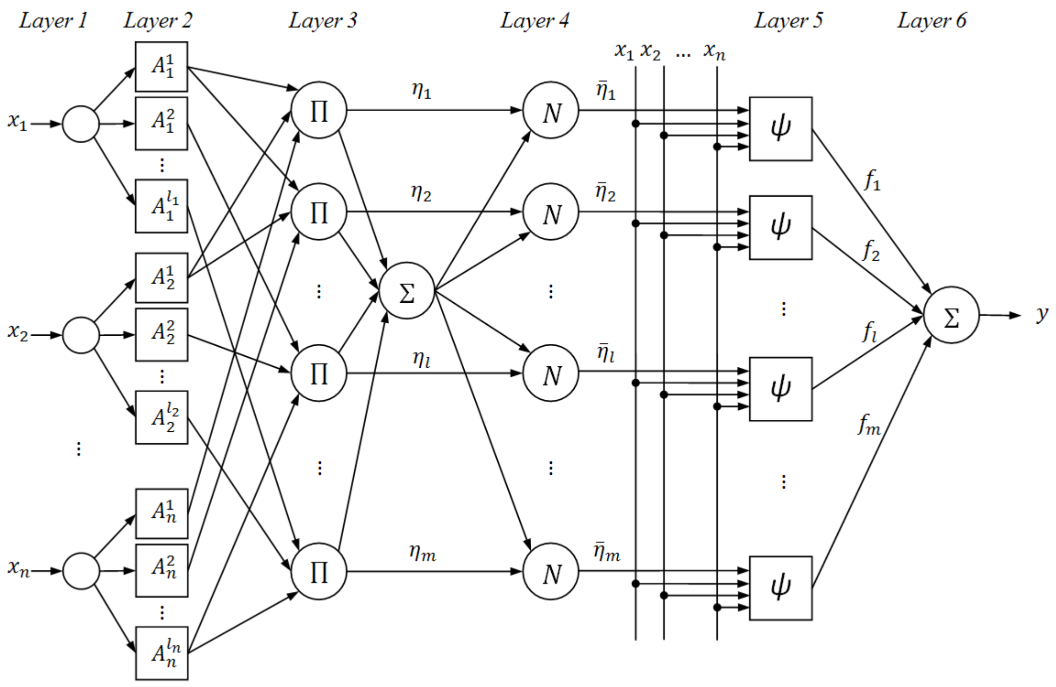

Figure 4.2. The structure of Fuzzy Wavelet Neural Network

- ***Layer 1:*** This layer is the input layer. Neurons in this layer simply cross the crisp input signals $x_1, x_2,\ldots,x_n$ to second layer.

- ***Layer 2:*** This layer is the fuzzification layer and neurons in this layer represent the fuzzy sets used in the antecedents of the fuzzy rules. A fuzzification neuron receives a crisp input and determines the degree to which this input belongs to the neuron's fuzzy set. The outputs of this layer are the values of membership functions for input values. There are total l_1 membership functions for the first input, l_2 for the second input and so on. Following equation shows i_jth Gaussian membership function for jth input variable.

$$A_j^{i_j} = \exp(-\frac{1}{2}(\frac{x_j - \mu_{i_j}}{\sigma_{i_j}})^2) \qquad j = 1,2,...,n \text{ and } i_j = 1,2,...,l_j \tag{4.4}$$

- ***Layer 3:*** This layer is the fuzzy rule layer. Each node in this layer represents a fuzzy rule. In order to calculate the firing strength of each rule, multiplication is used as AND (t-norm) operator. The output of the lth node in this layer is

$$\eta_l = \prod_{j=1}^{n} A_j^{i_j}(x_j) \tag{4.5}$$

where $l = i_1 i_2 \ldots i_n$, $i_1 = 1,\ldots,l_1$, $i_2 = 1,\ldots,l_2,\ldots,i_n = 1,\ldots,l_n$

Each possible combination of membership functions of all inputs represents a fuzzy rule. For example, lth rule in Figure 4.2 comprises from second membership functions of first and second inputs and last membership function of the last input. Thus total number of rules m is given by:

$$m = \prod_{i=1}^{n} l_i \tag{4.6}$$

- ***Layer 4:*** This layer is the normalization layer. Each neuron in this layer calculates the normalized activation strength of lth rule by:

$$\bar{\eta}_l = \frac{\eta_l}{\sum_{i=1}^{m} \eta_i} \qquad (l = 1,\ldots,m) \tag{4.7}$$

The normalized activation strength is the ratio of the activation strength of a given combination to the sum of activation strengths of all combinations. It represents the contribution of a given combination to the final result.

- ***Layer 5:*** This layer calculates the weighted consequent value of a given rule as follows for *l = 1,…,m*:

$$f_l = \overline{\eta_l} \Psi_l \tag{4.8}$$

For FWNN-S:

$$\Psi_l = \sum_{i=1}^{n} w_{il} (1 - (\frac{x_i - b_{il}}{c_{il}})^2) \exp(-\frac{1}{2}(\frac{x_i - b_{il}}{c_{il}})^2) \tag{4.9}$$

For FWNN-M:

$$\Psi_l = w_l \prod_{i=1}^{n} (1 - (\frac{x_i - b_{il}}{c_{il}})^2) \exp(-\frac{1}{2}(\frac{x_i - b_{il}}{c_{il}})^2) + p_l \tag{4.10}$$

For FWNN-R:

$$\Psi_l = w_l (1 - (\frac{\| \boldsymbol{x} - \boldsymbol{b}_l \|}{c_l})^2) \exp(-\frac{1}{2}(\frac{\| \boldsymbol{x} - \boldsymbol{b}_l \|}{c_l})^2) + p_l \tag{4.11}$$

- ***Layer 6:*** This layer contains only a single node and it computes the overall output as the summation of all incoming signals, which is given by:

$$y = \sum_{l=1}^{m} f_l \tag{4.12}$$

In [15]-[22], some fuzzy wavelet neural network models have been designed. The details of these models are explained in first chapter. The FWNN proposed in [21] also uses summation of dilated and translated versions of wavelet functions in consequent part of fuzzy rules for system identification and control purposes. However, there are some differences between the model in [21] and the FWNN-S described in this thesis. In [21], the number of fuzzy membership functions for each input is equal to the number of fuzzy rules, but in this thesis each input can have different number of membership functions and each possible combination of membership functions of inputs represents a fuzzy rule. The total number of rules can be calculated using (4.6). In [21], minimum is used as t-norm AND operator, but multiplication is used here for that purpose. In [21], summation of wavelet functions is multiplied by a weight. But, the effect of each input variable is different in overall output. Therefore, different weights are used for each single dimensional wavelet functions.

4.3 Training Algorithm for Fuzzy Wavelet Neural Network Models

The FWNN training is to encapsulate a given function or input-output pairs by adjusting network parameters. Unknown parameters of the FWNN models are adjusted by using BFGS gradient method. This method is explained in the third chapter. Unknown parameters are center parameters (***μ***) and scaling parameters (***σ***) of Gaussian membership functions in antecedent part of the rules, and translation (***b***), dilation (***c***) parameters of wavelet functions and weight (***w***) and bias (***p***) parameters in the consequent part of the rules. The number of these unknown parameters is given in Table 4.1 for each FWNN model.

Table 4.1. The number of unknown parameters for the FWNN models

Parameters / Models	***μ***	***σ***	***b***	***c***	***w***	***p***
FWNN-S	$\sum_{i=1}^{n} l_i$	$\sum_{i=1}^{n} l_i$	nm	nm	nm	0
FWNN-M	$\sum_{i=1}^{n} l_i$	$\sum_{i=1}^{n} l_i$	nm	nm	m	m
FWNN-R	$\sum_{i=1}^{n} l_i$	$\sum_{i=1}^{n} l_i$	nm	m	m	m

Initial values for scaling parameters of Gaussian membership functions and dilation parameters of wavelet functions are set to 1. Initial values for other parameters are generated randomly between 0 and 1.

4.3.1. Gradients of unknown parameters for FWNN models

While using a gradient based training algorithm like BFGS, gradients of the performance index with respect to all unknown parameters are required. The gradients of wavelet functions in consequent part of the rules for the FWNN-S can be calculated by the following formulas for $i = 1 \ldots, n$ and $l = 1, \ldots, m$:

$$\frac{\partial E}{\partial w_{il}}=\frac{\partial E}{\partial y}\bar{\eta}_l\left(1-\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right) \tag{4.13}$$

$$\frac{\partial E}{\partial b_{il}}=\frac{\partial E}{\partial y}\bar{\eta}_l w_{il}\frac{\left(x_i-b_{il}\right)}{c_{il}^2}\left(3-\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right) \tag{4.14}$$

$$\frac{\partial E}{\partial c_{il}}=\frac{\partial E}{\partial y}\bar{\eta}_l w_{il}\frac{\left(x_i-b_{il}\right)^2}{c_{il}^3}\left(3-\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right) \tag{4.15}$$

$$\frac{\partial E}{\partial y}=\frac{2}{N}\sum_{k=1}^{N}(y-y_d) \tag{4.16}$$

The gradients of Gaussian type membership functions in the antecedent part of the rules for the FWNN-S can be calculated by the following formulas for $j=1\ldots,n$ and $i_j=1,\ldots,l_j$:

$$\frac{\partial E}{\partial \mu_{i_j}}=\frac{\partial E}{\partial y}\frac{\partial y}{\partial A_j^{i_j}}\frac{(x_j-\mu_{i_j})}{\sigma_{i_j}^2}\exp\left(-\frac{1}{2}\left(\frac{x_j-\mu_{i_j}}{\sigma_{i_j}}\right)^2\right) \tag{4.17}$$

$$\frac{\partial E}{\partial \sigma_{i_j}}=\frac{\partial E}{\partial y}\frac{\partial y}{\partial A_j^{i_j}}\frac{(x_j-\mu_{i_j})^2}{\sigma_{i_j}^3}\exp\left(-\frac{1}{2}\left(\frac{x_j-\mu_{i_j}}{\sigma_{i_j}}\right)^2\right) \tag{4.18}$$

Here for the above calculations, partial derivative of the output y with respect to membership functions of each input variable is needed. For the first variable, this can be calculated as:

$$\frac{\partial y}{\partial A_1^{i_1}(x_1)}=\frac{\sum\limits_{i_2=1}^{l_2}\cdots\sum\limits_{i_n=1}^{l_n}\Psi_{i_1,i_2,\ldots,i_n}\prod\limits_{k=2}^{n}A_k^{i_k}(x_k)-y\sum\limits_{i_2=1}^{l_2}\cdots\sum\limits_{i_n=1}^{l_n}\prod\limits_{k=2}^{n}A_k^{i_k}(x_k)}{\sum\limits_{i_1=1}^{l_1}\sum\limits_{i_2=1}^{l_2}\cdots\sum\limits_{i_n=1}^{l_n}\prod\limits_{k=1}^{n}A_k^{i_k}(x_j)} \tag{4.19}$$

where $i_1=1,\ldots,l_1$.

The calculations can be easily generalized for the other input variable membership functions. The gradients of membership functions for FWNN-R and

FWNN-M models are same with gradients for FWNN-S model. Therefore, these gradients can be calculated using (4.17), (4.18) and (4.19). However, the gradients of wavelet functions for FWNN-R and FWNN-M models are different from those of FWNN-S model. These gradients for FWNN-R can be calculated using following formulas for *i =1…,n* and *l =1,…,m*:

$$\frac{\partial E}{\partial w_l}=\frac{\partial E}{\partial y}\bar{\eta}_l\left(1-\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^2}\right)\exp\left(-\frac{1}{2}\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^2}\right) \quad (4.20)$$

$$\frac{\partial E}{\partial b_{il}}=\frac{\partial E}{\partial y}\bar{\eta}_l w_l\frac{\left(x_i-b_{il}\right)}{c_l^2}\left(3-\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^2}\right)\exp\left(-\frac{1}{2}\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^2}\right) \quad (4.21)$$

$$\frac{\partial E}{\partial c_l}=\frac{\partial E}{\partial y}\bar{\eta}_l w_l\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^3}\left(3-\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^2}\right)\exp\left(-\frac{1}{2}\frac{\|\boldsymbol{x}-\boldsymbol{b_l}\|^2}{c_l^2}\right) \quad (4.22)$$

$$\frac{\partial E}{\partial p_l}=\frac{\partial E}{\partial y}\bar{\eta}_l \quad (4.23)$$

where $\|\boldsymbol{x}-\boldsymbol{b_l}\|^2=\sum_{i=1}^{n}(x_i-b_{il})^2$

The gradients for FWNN-M model can be calculated as for *i =1…,n* and *l=1,…,m*:

$$\frac{\partial E}{\partial w_l}=\frac{\partial E}{\partial y}\bar{\eta}_l\prod_{k=1}^{n}\phi_{kl} \quad (4.24)$$

$$\frac{\partial E}{\partial b_{il}}=\frac{\partial E}{\partial y}\bar{\eta}_l w_l\frac{\prod_{k=1}^{n}\phi_{kl}}{\phi_{il}}\frac{\left(x_i-b_{il}\right)}{c_{il}^2}\left(3-\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right) \quad (4.25)$$

$$\frac{\partial E}{\partial c_{il}}=\frac{\partial E}{\partial y}\bar{\eta}_l w_l\frac{\prod_{k=1}^{n}\phi_{kl}}{\phi_{il}}\frac{\left(x_i-b_{il}\right)^2}{c_{il}^3}\left(3-\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_i-b_{il}}{c_{il}}\right)^2\right) \quad (4.26)$$

$$\frac{\partial E}{\partial p_l}=\frac{\partial E}{\partial y}\bar{\eta}_l \quad (4.27)$$

where $\phi_{kj}=\left(1-\left(\frac{x_k-b_{kj}}{c_{kj}}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x_k-b_{kj}}{c_{kj}}\right)^2\right)$

4.3.2. Gradients of unknown parameters for SISO FWNN models

For SISO FWNN models, the outputs of wavelet functions in the fifth layer of the models, the equations (4.9), (4.10) and (4.11), can be simplified as follows:

$$\Psi_i = w_i(1-(\frac{x-b_i}{c_i})^2)\exp(-\frac{1}{2}(\frac{x-b_i}{c_i})^2) \qquad \text{for FWNN-S} \qquad (4.28)$$

$$\Psi_i = w_i(1-(\frac{x-b_i}{c_i})^2)\exp(-\frac{1}{2}(\frac{x-b_i}{c_i})^2)+p_i \quad \text{for FWNN-M and FWNN-R} \qquad (4.29)$$

where *i=1,..,m* and *m* is the number of membership function for input variable. Then, the overall output of SISO FWNN can be calculated as:

$$y = \frac{\sum_{i=1}^{m}\Psi_i \exp(-\frac{1}{2}(\frac{x-\mu_i}{\sigma_i})^2)}{\sum_{i=1}^{m}\exp(-\frac{1}{2}(\frac{x-\mu_i}{\sigma_i})^2)} \qquad (4.30)$$

The gradient of center and scaling parameters in membership functions for all SISO FWNN models can be calculated as:

$$\frac{\partial E}{\partial \mu_i} = \frac{\partial E}{\partial y}\frac{\Psi_i - y}{\sum_{k=1}^{m}\exp\left(-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma_k}\right)^2\right)}\frac{(x-\mu_i)}{\sigma_i^2}\exp\left(-\frac{1}{2}\left(\frac{x-\mu_i}{\sigma_i}\right)^2\right) \qquad (4.31)$$

$$\frac{\partial E}{\partial \sigma_i} = \frac{\partial E}{\partial y}\frac{\Psi_i - y}{\sum_{k=1}^{m}\exp\left(-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma_k}\right)^2\right)}\frac{(x-\mu_i)^2}{\sigma_i^3}\exp\left(-\frac{1}{2}\left(\frac{x-\mu_i}{\sigma_i}\right)^2\right) \qquad (4.32)$$

The gradients of wavelet function parameters can be calculated as follows for SISO FWNN models.

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y}\bar{\eta}_i\left(1-\left(\frac{x-b_i}{c_i}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x-b_i}{c_i}\right)^2\right) \qquad (4.33)$$

$$\frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial y}\bar{\eta}_i w_i\frac{(x-b_i)}{c_i^2}\left(3-\left(\frac{x-b_i}{c_i}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x-b_i}{c_i}\right)^2\right) \qquad (4.34)$$

$$\frac{\partial E}{\partial c_i} = \frac{\partial E}{\partial y}\overline{\eta}_i w_i \frac{(x-b_i)^2}{c_i^3}\left(3-\left(\frac{x-b_i}{c_i}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{x-b_i}{c_i}\right)^2\right) \tag{4.35}$$

$$\frac{\partial E}{\partial p_i} = \frac{\partial E}{\partial y}\overline{\eta}_i \quad \text{for FWNN-M and FWNN-R} \tag{4.36}$$

where $i=1,..,m$ and $\overline{\eta}_i = \dfrac{\exp\left(-\frac{1}{2}\left(\frac{x-\mu_i}{\sigma_i}\right)^2\right)}{\sum_{k=1}^{m}\exp\left(-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma_k}\right)^2\right)}$ for SISO FWNN models

5. SIMULATION RESULTS

The proposed AWN and FWNN models are applied to six well-known benchmark problems to compare the performance of these models with existing models. These problems are a SISO function approximation, Box-Jenkins, Mackey Glass, and sunspot number time series predictions and system identification of two nonlinear plants.

5.1. Approximation of a Piecewise Function

A piecewise function studied by Zhang [7] and Chen [27] is used to compare the FWNN and AWN models with some other wavelet-based networks. This function is defined as

$$y_d = f(x) = \begin{cases} -2.186x - 12.864 & -10 \le x < -2 \\ 4.246x & -2 \le x < 0 \\ 10e^{-0.05x-0.5}\sin[(0.03x+0.7)x] & 0 \le x \le 10 \end{cases} \tag{5.1}$$

For the training process, N = 200 sample points are drawn from the data uniformly distributed over [-10, 10]. In order to compare the proposed models with other works, the measure in [15] is used:

$$J = \sqrt{\frac{\sum_{i=1}^{N}(y-y_d)^2}{\sum_{i=1}^{N}(y_d-y_{av})^2}} \tag{5.2}$$

where y_d is actual output, y is predicted output and $y_{av} = \frac{1}{N}\sum_{i=1}^{N} y_d$.

In Table 5.1, it is seen that the performance of the first order AWN with 8 rules and FWNN-M (or FWNN-R) with 7 rules is superior to that of the other WNNs. Figure 5.1 illustrates the validity of first order AWN with hybrid algorithm which corresponds the smallest performance measure value among the simulations.

Table 5.1. Comparison of AWN and FWNN models for the piecewise function

Model	Number of parameters	J
FWNN-S	35	0.0057
FWNN-S	30	0.0114
FWNN-M and FWNN-R	42	0.0031
FWNN-M and FWNN-R	36	0.0041
AWN-F	32	0.0033
AWN-F	28	0.0047
AWN-Z	24	0.0088
AWN-Z	21	0.0371
FWN [15]	28	0.021
WNN[7]	22	0.05057
WNN[27]	23	0.0480

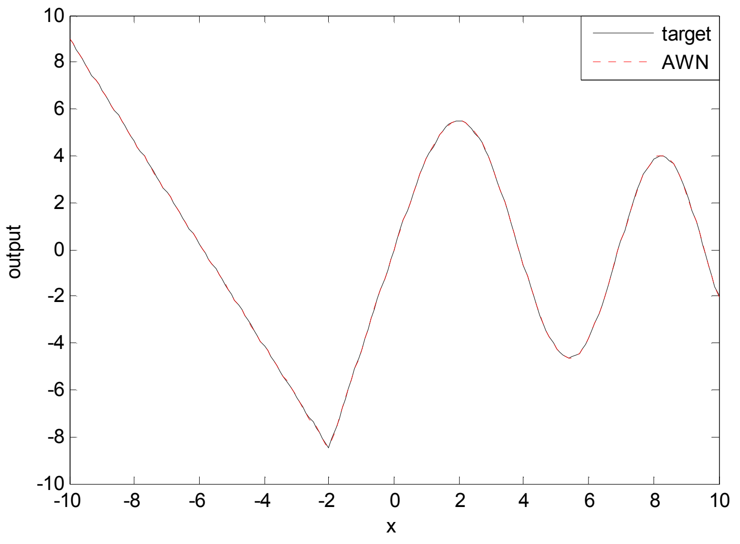

Figure 5.1. Actual and predicted values with first order AWN for the piecewise function

5.2. Prediction of Box-Jenkins Time Series

In this section, the proposed models are applied to Box-Jenkins time series data (gas furnace data) which was recorded from a combustion process of a methane-air mixture. The input of this process is the gas flow rate u(t) and the output y(t) is the CO_2 concentration in outlet gas. In order to predict y(t), u(t-4) and y(t-1) are used as inputs to models. The original dataset includes 296 input-output pairs. However, it is reduced to 292 pairs because of delay of inputs. The first 200 input-output pairs are used as a training set, and the remaining 92 points are used as a test set to see the prediction performance AWN and FWNN models.

These models are trained and tested for both two and three membership functions for each input, which form 4 and 9 fuzzy rules respectively. Each FWNN model is trained for 500 epochs and each AWN model is trained for 400 epochs. Figure 5.2 and Figure 5.3 show the actual time series with the output of FWNN-M with three membership functions for each input and the prediction error, respectively. Root mean square error (RMSE)

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y - y_d)^2} \tag{5.3}$$

is used as performance measure in this example. For the best result, a comparison of the proposed models is summarized in Table 5.2 with respect to RMSE values for training and testing. In Table 5.3, the FWNN models are compared with different models in the literature. It is seen from the training and testing performances that FWNN-M model gives the second best result and LLWNN[9]+hybrid model is the best of the others. This model has 56 learning parameters. Our FWNN models have its best results with 66 learning parameters in testing. In addition, the number of training data of Box-Jenkins time series consists of only 200 input-output pairs that result in overtraining in testing the predictions of the models. This means that it is necessary to increase model parameters or Box-Jenkins time series data does not represent the general characteristic of this process. To see the prediction performance and to show the efficiency of the proposed models, new simulation experiments are done such as Mackey Glass time series prediction and system identification of two nonlinear plants. As you will see, the results of these new simulation examples prove both conclusions about the Box-Jenkins time series data prediction. To obtain the best accurate models, the number of neurons in the hidden layer, i.e. the number of fuzzy rules should be increased such that an optimal increase in the number of training parameters can be obtained.

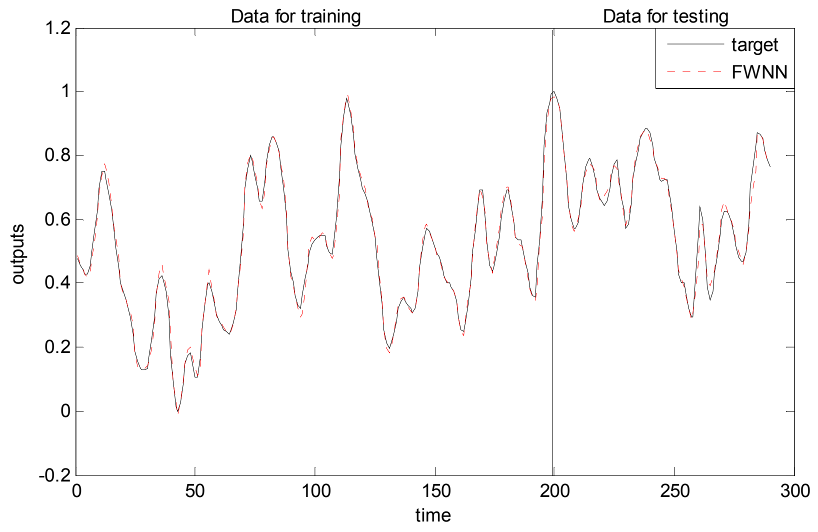

Figure 5.2. Actual and predicted values with FWNN-M for Box-Jenkins time series

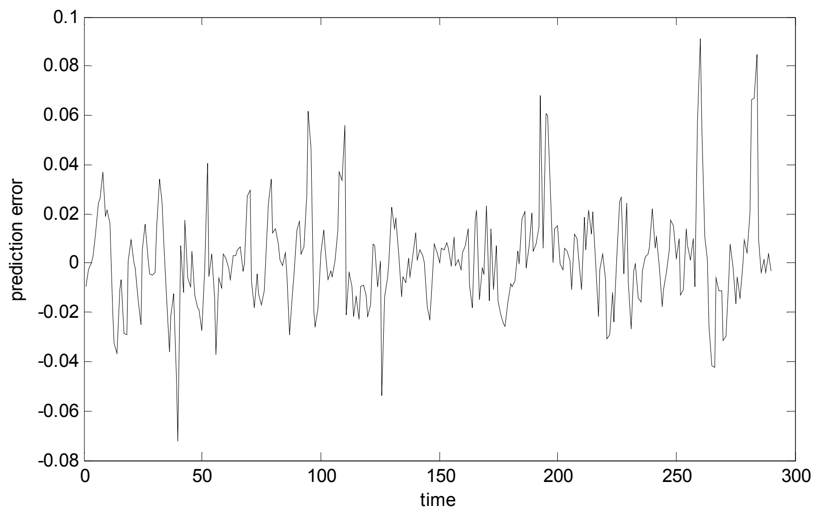

Figure 5.3. Prediction error of the FWNN-M model for Box-Jenkins time series

Table 5.2. Comparison of AWN and FWNN models for Box Jenkins time series

Model	**Number of parameters**	**Epoch**	**RMSE training**	**RMSE testing**
AWN-Z (2 MFs)	12	400	0.02153	0.03308
AWN-Z (3 MFs)	21	400	0.02003	0.03192
AWN-F (2 MFs)	20	400	0.01934	0.03176
AWN-F (3 MFs)	39	400	0.01909	0.03084
FWNN-S (2 MFs)	32	500	0.01884	0.03085
FWNN-S (3 MFs)	66	500	0.01880	0.02778
FWNN-R (2 MFs)	28	500	0.01992	0.03171
FWNN-R (3 MFs)	57	500	0.01881	0.02794
FWNN-M(2 MFs)	32	500	0.01900	0.02963
FWNN-M(3 MFs)	66	500	0.01963	0.02324

Table 5.3. Comparison of test results of different models for Box Jenkins time series

Model	RMSE
Tong's model [28]	0.685
Pedrycz's model [29]	0.566
Xu's model [30]	0.573
Sugeno's model [31]	0.596
Surmann's model [32]	0.400
FuNN model [33]	0.071
HyFIS model [34]	0.042
Neural tree model [35]	0.026
WNN[9]+gradient	0.084
WNN[9]+hybrid	0.081
LLWNN[9]+gradient	0.017
LLWNN[9]+hybrid	0.013
AWN-Z (3 MFs)	0.032
AWN-F (3 MFs)	0.031
FWNN-S (3 MFs)	0.028
FWNN-R (3 MFs)	0.028
FWNN-M (3 MFs)	0.023

5.3. Prediction of Sunspot Number Time Series

In this section, annually recorded sunspot time series for the years 1700-1979 is considered to show performance of the AWN and FWNN models. These numbers show the yearly average relative number of sunspots observed. To make the comparisons meaningful with other works, the dataset is divided into three parts. The data points between years 1700-1920 are used for training the models. The data points for years 1921-1955 and 1956-1979 form first and second test sets respectively. The y(t-4), y(t-3), y(t-2) and y(t-1) are used as inputs to models in order to predict the output y(t). Two membership functions are selected for each input, so there are total 16 rules in each model and these models are trained for 200 epochs. Normalized mean square error (NMSE)

$$NMSE = J^2 \tag{5.4}$$

is used to compare the proposed AWN and FWNN with other models, where J equals (5.2). Training and testing error values are given in Table 5.4 with comparison of other models in the literature. In Figure 5.4 and Figure 5.5, actual output of time series, prediction results of summation FWNN and prediction error are shown.

Table 5.4. Comparison of AWN and FWNN models with different models for sunspot number time series

Model	Number of parameters	NMSE training	NMSE testing 1	NMSE testing 2
Tong and Lim [36]	16	0.097	0.097	0.28
Weigend [37]	43	0.082	0.086	0.35
Svarer [38]	12-16	0.090	0.082	0.35
Transversal Net[39]	14	0.0987	0.0971	0.3724
Recurrent net[39]	22	0.1006	0.0972	0.4361
AWN-Z	32	0.1093	0.2101	0.1734
AWN-F	96	0.1225	0.1447	0.1468
FWNN-S	208	0.0895	0.1093	0.1510
FWNN-R	128	0.0796	0.1099	0.2549
FWNN-M	176	0.0828	0.0973	0.1988

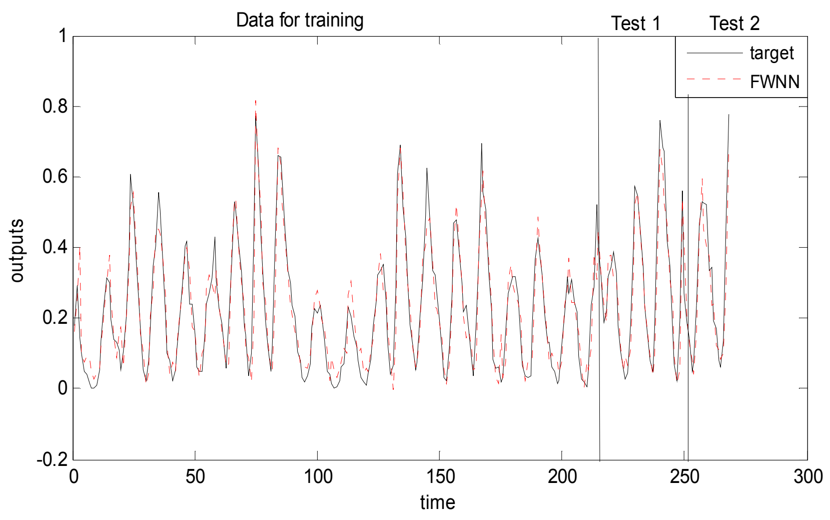

Figure 5.4. Actual and predicted values with FWNN-S model for sunspot number time series

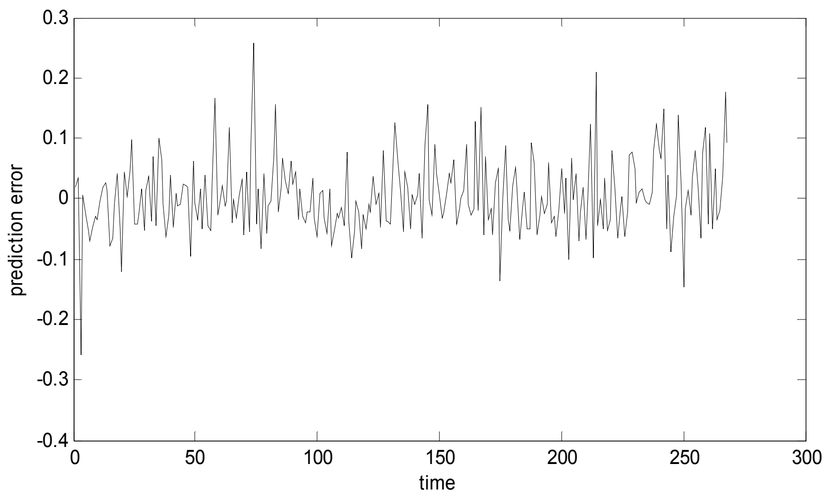

Figure 5.5. Prediction error of the FWNN-S model for sunspot number time series

5.4. Prediction of Mackey Glass Time Series

In this section, the proposed models are applied to Mackey Glass time series. This is a benchmark chaotic time series and a widely investigated problem in the neuro-fuzzy literature. The time series is generated by the following differential equation:

$$\frac{dx}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{5.5}$$

For τ = 17 the systems response is chaotic and simulation data is obtained using initial conditions x(0)=1.2 and τ = 17. 1000 input-output data points are extracted from the Mackey Glass time series x(t) where t=118 to 1117. x(t+6) is predicted using the input variables x(t), x(t-6), x(t-12) and x(t-18). The first 500 data points are used to train the models and remaining 500 points are used for validating the identified model. Total 16 fuzzy rules are generated using two membership functions for each input variable. The comparison of the models proposed in this thesis is illustrated in Table 5.5. Also, test results for different models are summarized in Table 5.6. Figure 5.6 and Figure 5.7 show the actual time series with the output of the FWNN-S and the prediction error, respectively.

Table 5.5. Comparison of AWN and FWNN models for Mackey-Glass time series

Model	**Number of parameters**	**Epoch**	**RMSE training**	**RMSE testing**
AWN-Z	32	2000	0.00992	0.00982
AWN-F	96	4000	0.00183	0.00178
FWNN-S	208	5000	0.00124	0.00109
FWNN-R	128	5000	0.00231	0.00232
FWNN-M	176	5000	0.00129	0.00114

Table 5.6. Comparison of test results of different models for Mackey-Glass time series

Model	RMSE
Auto-regressive model	0.19
Cascade correlation NN	0.06
Backpropagation NN	0.02
Sixth-order polynomial	0.04
Linear prediction method	0.55
Product T-norm [40]	0.09
Classical RBF (with 23 neurons) [41]	0.0114
PG-RBF network [42]	0.0028
Genetic algorithm and fuzzy system [43]	0.049
Neural tree model [35]	0.0069
Radial basis function network [44]	0.0015
WNN [9] + gradient	0.0071
WNN [9]+ hybrid	0.0059
LLWNN [9] + gradient	0.0041
LLWNN [9] + hybrid	0.0036
AWN-Z	0.00982
AWN-F	0.00178
FWNN-S	0.00109
FWNN-R	0.00232
FWNN-M	0.00114

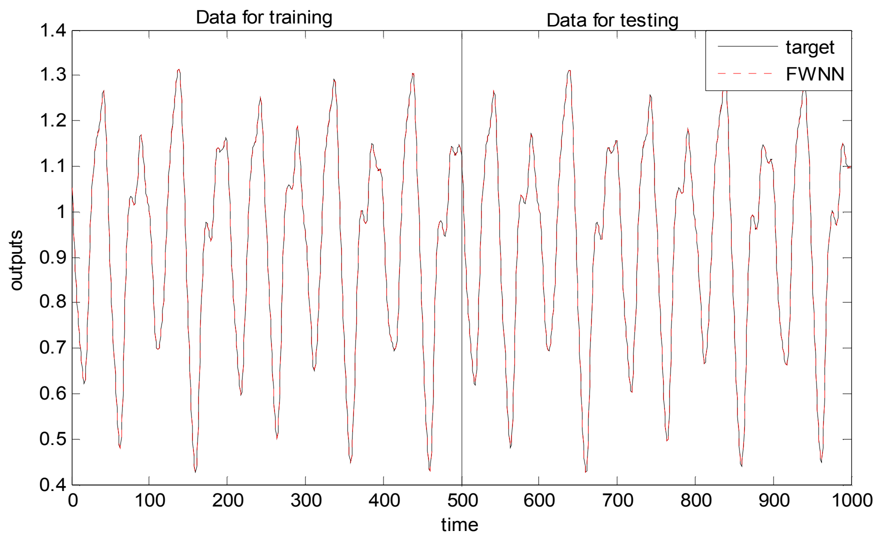

Figure 5.6. Actual and predicted values with FWNN-S model for Mackey-Glass time series

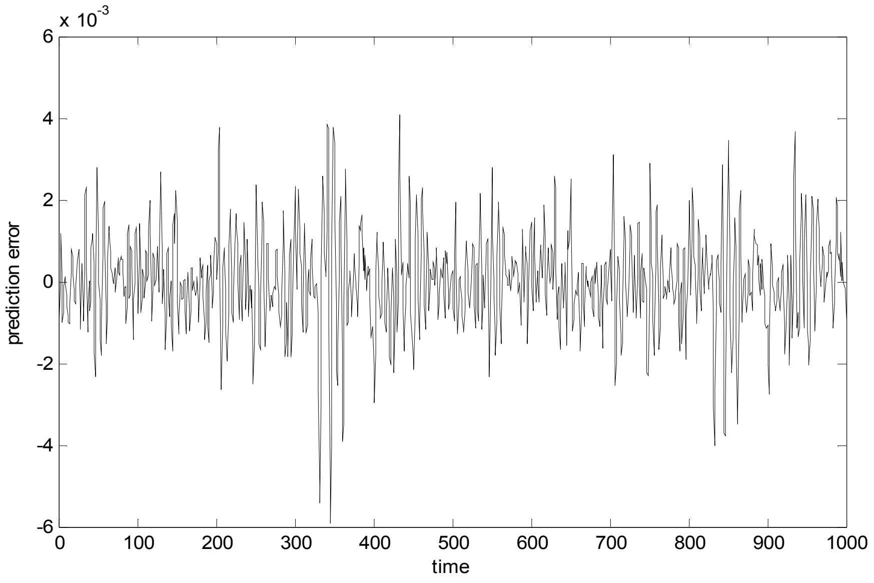

Figure 5.7. Prediction error of the FWNN-S model for Mackey-Glass time series

As can be seen, the RMSE values of the proposed FWNN models are much less than the other models, but models have considerably larger number of parameters to be learned.

5.5. System Identification Example 1

System identification involves finding the relation between the input and output of the system [21, 45]. The structure of series-parallel system identification model with AWN or FWNN is shown in Figure 5.8. The inputs of the model are delayed values of control signal u(k) and output of the plant y(k). Here, y(k) is target output of plant and $y_p(k)$ is predicted output. AWN or FWNN model parameters are updated according to prediction error e(k).

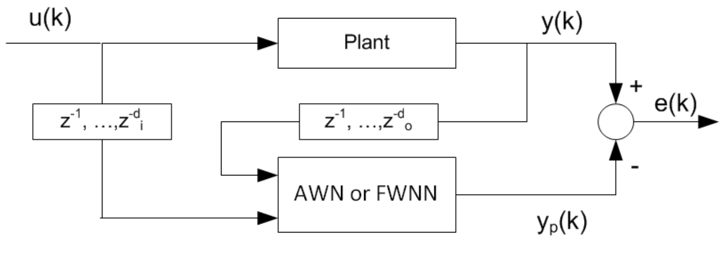

Figure 5.8. Series-Parallel identification model with AWN or FWNN

In this example, the plant to be identified is given by following equation:

$$y(k) = 0.72y(k-1) + 0.025y(k-2)u(k-2) + 0.01u^2(k-3) + 0.02u(k-4) \tag{5.6}$$

The output of the system depends on two previous output values and three previous input values. However, only u(k-1) and y(k) are used as inputs to the models to predict y(k+1). Two membership functions are selected for each input of the models. In order to train models, 900 inputs are used similar to the inputs used in [47] and [48]. The half of the inputs is independent and identically distributed (i.i.d.) uniform sequence over [-2, 2] and the remaining is a sinusoid given by 1.05sin(πk/45). AWN models are trained for 100 epochs and FWNN models are trained for 200 epochs. After training, the following input signal which is same test signal with other compared models is used for testing the performance of the models.

$$u(k) = \begin{cases} \sin(\pi k/25) & k < 250 \\ 1.0 & 250 \le k < 500 \\ -1.0 & 500 \le k < 750 \\ 0.3\sin(\pi k/25) + 0.1\sin(\pi k/32) & \\ \quad + 0.6\sin(\pi k/10) & 750 \le k < 1000 \end{cases} \tag{5.7}$$

Figure 5.9 shows the actual and predicted output of the plant for test signal with zero order AWN model. From Table 5.7, it can be seen that the proposed AWN and FWNN models illustrate much better performance than the other models in this system identification problem.

Table 5.7. Comparison of AWN and FWNN models with other models for system identification example 1

Models	Network Parameters	RMSE Training	RMSE Testing
ERNN[46]	54	0.036	0.078
RSONFIN[47]	49	0.03	0.06
TRFN-S[48]	33	0.0067	0.0313
FWNN[21]	27	0.019736	0.022609
FWNN[21]	43	0.018713	0.020169
AWN-Z	12	0.009368	0.022933
AWN-F	20	0.009391	0.023259
FWNN-S	32	0.009771	0.022226
FWNN-R	28	0.009688	0.022204
FWNN-M	32	0.009635	0.021342

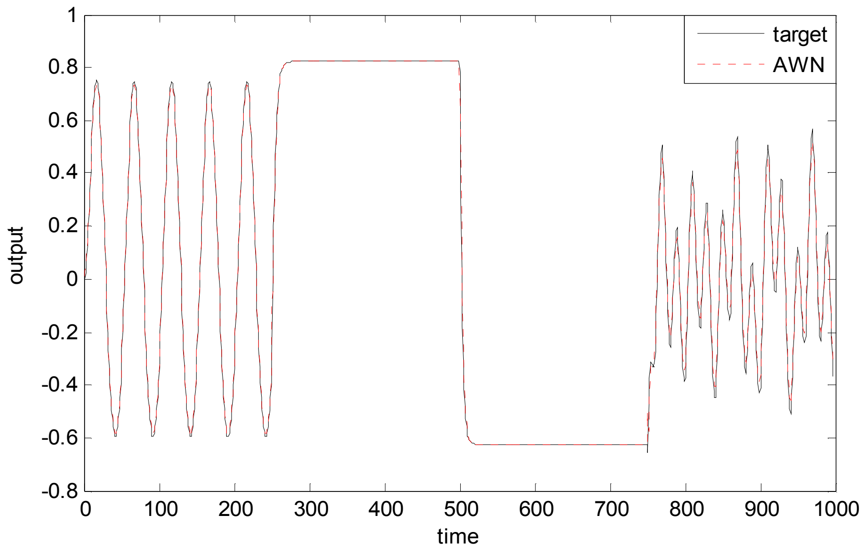

Figure 5.9. Actual and predicted test signal values with AWN-Z model for system identification example 1

5.6. System Identification Example 2

This example considers modeling the nonlinear plant given by following equation.

$$y(k+1) = f(y(k), y(k-1), y(k-2), u(k), u(k-1)) \tag{5.8}$$

where

$$f(x_1, x_2, x_3, x_4, x_5) = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_3^2 + x_2^2} \tag{5.9}$$

The current output of the plant depends on three previous output values and two previous input values. However, we use only y(k) and u(k) to predict y(k+1). In order to train the models, 900 training inputs are generated as in previous example. The number of membership functions is also same with previous example. To test the models for this plant, the test signal in (5.7) is used. The actual and predicted values for testing with radial FWNN are shown in Figure 5.10. As it is seen in Table 5.8, the FWNN models with smaller parameters are successful in identification than the compared models in the literature.

Table 5.8. Comparison of AWN and FWNN models with other models for system identification example 2

Models	**Network Parameters**	**RMSE Training**	**RMSE Testing**
RFNN[49]	112	0.0114	0.0575
RSONFIN[47]	36	0.0248	0.0780
Feedforw. Neur. Fuz. Sys.[48]	48	0.0203	0.0521
TRFN-S[48]	33	0.0084	0.0346
FWNN[21]	27	0.029179	0.031212
FWNN[21]	43	0.028232	0.030125
AWN-Z	12	0.023496	0.037763
AWN-F	20	0.025174	0.040632
FWNN-S	32	0.020888	0.033724
FWNN-R	28	0.015274	0.032116
FWNN-M	32	0.019269	0.033327

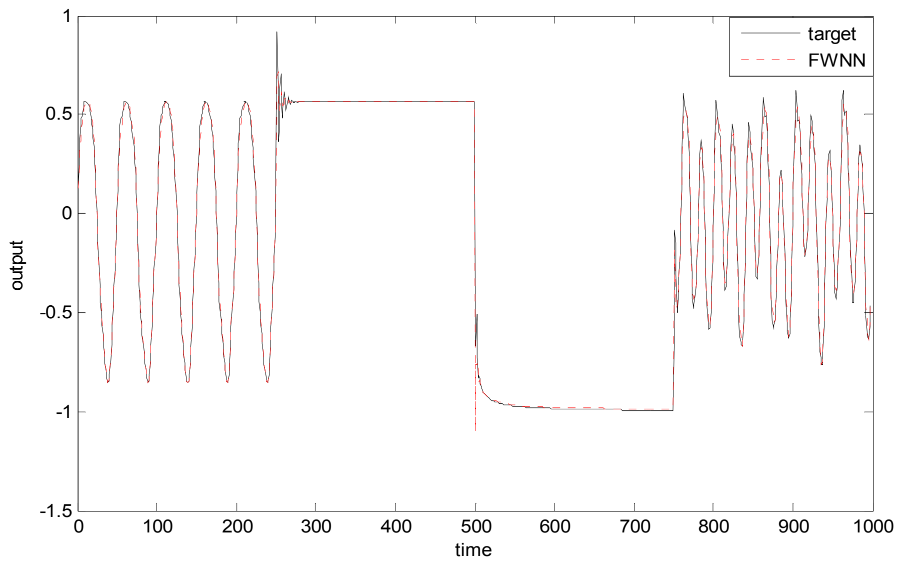

Figure 5.10. Actual and predicted test signal values with FWNN-R model for system identification example 2

6. CONCLUSION

In this thesis, two new type of neuro-fuzzy models which use wavelet basis functions in their processing units are introduced and are used for time series prediction and system identification problems. Wavelet functions are firstly used in antecedent part of the fuzzy rules and zero and first order polynomial functions are used in consequent parts. Secondly, wavelets are used in consequent part of the rules and Gaussian functions are used as membership functions in antecedent parts.

With same number of rules, FWNN models have more parameters to be determined than AWN models. In piecewise function approximation, first order AWN model gives better results with same number of rules although it has less model parameters. In Box-Jenkins, Mackey-Glass and sunspot time series prediction, FWNN models give better results with higher number of parameters. While FWNN models give smaller error values in second system identification problem, both FWNN and AWN models give close results in first system identification example.

In addition, all of these models are compared with other models in the literature. In piecewise function approximation, first order AWN model gives best result among other models in the literature with same and less number of parameters. In sunspot time series prediction, we have better results with proposed FWNN models. However these models have higher number of parameters than other models. In Mackey-Glass time series prediction, the FWNN-S and FWNN-M models give the best results. Some other models in the literature give better results than proposed models in this thesis for Box-Jenkins time series prediction. In first system identification example, AWN and FWNN models give smaller training error values and close testing error values among other models with less parameters. In second system identification example, the FWNN models give close test results among other models with same and less parameters in other models.

It is believed that these models can also be applied to a wider range of real-world problems such as speech and image processing, financial data analysis and prediction and other system identification and control applications. In addition, the significance of this work is to show that an AWN or FWNN model can track any nonlinear dynamical function. Secondly, efficient computational models and algorithms can be designed for parameter identification in fully nonlinear systems in general and for training AWN or FWNN models as a specific application. An approximate second order gradient procedure has been used here. Other optimization techniques such as particle swarm optimization (PSO) or some hybrid algorithms which combines gradient based algorithms with PSO can also be used for training unknown parameters of the models.

REFERENCES

[1] Krose B., and Smagt P., *An Introduction to Neural Networks*, University of Amsterdam, 8th Edition, 1996.

[2] Haykin S., *Neural Networks – A Comprehensive Foundation*, Pearson Education, 2nd Edition, Upper Saddle River, N.J., 1999.

[3] Valens C., *A Really Friendly Guide to Wavelets*, 1999. http://pagesperso-orange.fr/polyvalens/clemens/download/arfgtw.pdf

[4] Sitharama Iyengar S., Cho E.C., and Phoha V. V. *Foundations of Wavelet Networks and Applications*, CRC Press, Inc. Boca Raton, FL, USA, 2002.

[5] Graps A., "An Introduction to Wavelets", *IEEE Computational Science and Engineering*, **2(2)**, 50-61, 1995.

[6] Zhang Q., "Using wavelet networks in nonparametric estimation", *IEEE Transactions on Neural Networks*, **8**, 227-336, 1997.

[7] Zhang Q., and Benveniste A., "Wavelet networks", *IEEE Trans. Neural Netw.*, **3**, 889-898, 1992.

[8] Zhang J., Walter G.G., and Lee W.N.W., "Wavelet neural networks for function learning", *IEEE Trans. Signal Process.*, **43**, 1485-1497, 1995.

[9] Chen Y., Yang B., and J. Dong, "Time-series prediction using a local linear wavelet neural network", *Neurocomputing*, **69**, 449-465, 2006.

[10] Galvão R.K.H., Becerra V. M., Calad J. M.F., and Sılva P. M., "Linear–Wavelet Networks", *Int. J. Appl. Math. Comput. Sci.*, **14(2)**, 221–232, 2004.

[11] Billings S. A., and Wei H. L., **"**A New Class of Wavelet Networks for Nonlinear System Identification", *IEEE Trans. Neural Netw.*, **16(4)**, 2005.

[12] Holmes C. C., and Mallick B. K., "Bayesian Wavelet Networks for Nonparametric Regression", *IEEE Trans. Neural Netw.*, **11(1)**, 2000.

[13] Fuller R., *Introduction to Neuro-Fuzzy Systems*, Advances in Soft Computing, Physica-Verlag Heidelberg, 2000.

[14] Jang J. S. R., "ANFIS: adaptive-network-based fuzzy inference systems", *IEEE Trans. Syst., Man, Cybern.*, **23(3)**, 665-685, 1993.

[15] Ho, D.W.C., Zhang P.A, and Xu J., "Fuzzy wavelet networks for function learning", *IEEE Trans. Fuzzy Syst.*, **9(1)**, 200-211, 2001.

[16] Zekri M., Sadri S., and Sheikholeslam F., “Adaptive fuzzy wavelet network control design for nonlinear systems”, *Fuzzy Sets and Systems,* **159**, 2668 – 2695, 2008.

[17] Karatepe E., and Alçı M., “A new approach to fuzzy wavelet system modeling”, *International Journal of Approximate Reasoning*, **40(3)**, 302-322, 2005.

[18] Banakar A., and Azeem M. F., “Artificial wavelet neural network and its application in neuro-fuzzy models”, *Applied Soft Computing*, **8**, 1463–1485, 2008.

[19] Wang Z., Peng H., and Wang J., “Research for a Dynamic Recurrent Fuzzy Wavelet Network”, *Proce. of the 6. Int. Conf. on Intelligent Systems Design and Applications*, vol:1, 914-918, 2006.

[20] Srivastavaa S., Singha M., Hanmandlub M., and Jha A.N., “New fuzzy wavelet neural networks for system identification and control”, *Applied Soft Computing*, **6**, 1–17, 2005.

[21] Abiyev R. H., and Kaynak O., “Fuzzy Wavelet Neural Networks for Identification and Control of Dynamic Plants-A Novel Structure and Comparative Study”, *IEEE Trans. Ind. Electron.*, **55(8)**, 3133 – 3140, 2008.

[22] Lin C. J., Chin C. C., and Lee C. L., “A wavelet-based neuro-fuzzy system and its applications”, *Proc. of the Int. Joint Conf. on Neural Networks*, vol:3, 1921- 1926, 2003.

[23] Jang J. S. R., Sun C. T., and Mizutani E., *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, Prentice Hall, 1997.

[24] Mallat S., “A Theory for Multiresulation Signal Decomposition: the Wavelet Representation”, *IEEE Trans. Pattern Anal. Mach. Intell.*, **11(7)**, 674-693, 1989.

[25] Sanner R., and Slotine J-J. E., “Gaussian Networks for direct Adaptive Control”, *IEEE Trans. Neural Netw.*, **13(6)**, 837-863, 1992.

[26] Gill P.E., Murray W., and Wright M.H., *Practical Optimization*, Academic Press Ltd., 1993.

[27] Chen J., and Bruns D.D., "WaveARX neural network development for system identification using a systematic design synthesis," *Ind. Eng. Chem. Res.*, **34**, 4420-4435, 1995.

[28] Tong R.M., "The evaluation of fuzzy models derived from experimental data", *Fuzzy Sets and Systems*, **4**, 1-12, 1980.

[29] Pedrycz W., "An identification algorithm in fuzzy relational systems", *Fuzzy Sets and Systems*, **13**, 153-167, 1984.

[30] Xu C.W., "Fuzzy model identification and self-learning for dynamic systems", *IEEE Trans. Syst., Man, Cybern.*, **17**, 683-689, 1987.

[31] Sugeno M., et al., "Linguistic modeling based on numerical data", in *Proc. IFSA'91*, 234-247, 1991.

[32] Surmann H., et al., "Self-organizing and genetic algorithm for an automatic design of fuzzy control and decision systems", in *Proc. FUFIT's 93*, 1079-1104, 1993.

[33] Kasabov N.K., et al., "FuNN/2-A fuzzy neural network architecture for adaptive learning and knowledge acquisition", *Information Science*, **101**, 155-175, 1997.

[34] Kim J., and Kasabov N., "HyFIS: adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems", *Neural Networks*, **12**, 1301-1319, 1999.

[35] Chen Y.H., et al., "Nonlinear system modeling via optimal design of neural trees", *Int. Journal of Neural Systems*, **14(2)**, 125-137, 2004.

[36] Tong H., and Lim K. S., "Threshold autoregression, limit cycles and cyclical data", *Journal Royal Statistical Society B*, **42**, 245–292, 1980.

[37] Weigned A. S., Rumelhart D. E., and Huberman B. A., *Predicting the future: A connectionist approach*, Techn. Rep. Stanford-PDP-90- 01 or PARC-SSL-90-20, 1990.

[38] Svarer C., Hansen L. K., and Larsen J., "On design and evaluation of tapped-delay neural network architectures", in *Proc. IEEE Int. Conf. Neural Netw.*, San Francisco ,1992.

[39] McDonnell J. R., and Waagen D., "Evolving Recurrent Perceptrons for Time-Series Modeling", *IEEE Trans. Neural Netw.*, **5(1)**, 24-38, 1994.

[40] Wang L.X., et al., "Generating fuzzy rules by learning from examples", *IEEE Trans. Syst., Man, Cybern.,* **22**, 1414–1427, 1992.

[41] Cho K.B., et al., "Radial basis function based adaptive fuzzy systems their application to system identification and prediction", *Fuzzy Sets and Systems*, **83**, 325–339, 1995.

[42] Rojas I., et al., "Time series analysis using normalized PG-RBF network with regression weights", *Neurocomputing*, **42**, 167–285, 2002.

[43] Kim D., et al., "Forecasting time series with genetic fuzzy predictor ensembles", *IEEE Trans. Fuzzy Syst.,* **5**, 523–535, 1997.

[44] Harpham C., Dawson C.W., "The effect of different basis functions on a radial basis function network for time series prediction: a comparative study", *Neurocomputing*, **69(16)**, 2161–2170, 2006.

[45] Narendra K. S., and Parthasarathy K., "Identification and control dynamical systems using neural networks", *IEEE Trans. Neural Netw.*, **1(1)**, 4–27, 1990.

[46] Elman J. L., "Finding structure in time", *Cognit. Sci.,* **14(2)**, 179–211, 1990.

[47] Juang C. F., and Lin C. T., "A recurrent self-organizing neural fuzzy inference network", *IEEE Trans. Neural Netw.*, **10(4)**, 828–845, 1999.

[48] Juang C.-F., "A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms", *IEEE Trans. Fuzzy Syst.*, **10(2)**, 155–170, 2002.

[49] Lee C. H., and Teng C. C., "Identification and control of dynamic systems using recurrent fuzzy neural networks," *IEEE Trans. Fuzzy Syst.*, **8**, 349–366, 2000.