Chapter 2
Adaptive Neuro-Fuzzy Interference System

Abstract This chapter explains in detail the theoretical background of Artificial Neural Network (ANN) and Adaptive Neuro-Fuzzy Inference System (ANFIS). The detailed explanation of this method will highlight its importance in the estimation of ZTD model.

Keywords Artificial neural network · ANFIS · Fuzzy inference system · Hybrid learning algorithm · Backpropagation

2.1 Artificial Neural Networks

Generally, an artificial neural network (ANN) is a system developed for information processing, where it has a similar way with the characteristics of biological neural systems. It was developed based on the human brain, which is capable of processing information, which are complex, nonlinear, and being able to work in parallel, distributed, and local processing and adaptation. ANN is designed to resemble the brain systems such as the construction of architectural structures, learning techniques, and operating techniques. This is the reason that ANN has been widely adopted by scientists because of its accuracy and its ability to develop complex nonlinear models and is used to solve a wide variety of tasks, especially in the field of climate and weather. This section will discuss the capabilities of ANN such as neurons modeling, architecture, and its learning process.

2.1.1 Neuron Modeling

In the human brain, there are neurons that are interconnected to one another. These neurons act as a tool that can perform processing of information of human senses. Haykin (2009) described that a biological neuron consists of a cell body, where
conditions are covered by the cell membrane (Fig. 2.1). Each cell has branches called dendrites. Dendritic play a role in receiving the information into the cells of the body through the axon.

The axon is a long single fiber that can carry the signal from the cell body toward the neuron—the next neuron. The meeting point between neurons with the next neuron found in a small space between dendrites and axons is known as a synapse. The space of synapses is applicable for shipping and receiving all information processes from the senses. Any information entered will be encoded in the form of electrical signals. All electrical signals into the synapses are counted and calculated. When the number of electrical signals regardless of the limits or thresholds specified in the synapse, the synapses react to a new electrical signal input to be used by the next neuron. If the electrical signals cannot be separated from the predetermined threshold, then the synapses will be retarded. Retardation of synapses causes obstruction of the relationship between the two neurons.

In line with the biological neuron model, McCulloch and Pitt (1943) proposed a model neuron that has the characteristics of the transmission and receipt of information process that is similar to the process that occurs in biological neurons. This neuron modeling was becoming a reference in the development of ANN model at current state. A neuron plays a role in determining the function and operation of the network. The mathematical models of neurons, which are commonly used in the ANN model is shown in Fig. 2.2.

Neuron modeling based on Fig. 2.2 can be represented by the following mathematical equation:

\[ u_{(k)} = \sum_{j=1}^{n} w_{kj} x_j \text{ and } y_{(k)} = \phi(u_{(k)}) + b_{(k)} \]  \hspace{1cm} (2.1)

where \( u_{(k)} \) is the output of the adder function neuron model, \( x_j \) is data or input signal on path synapse \( j \), and \( w_{kj} \) is the weighted in the path of synapse \( j \) to \( k \) neuron. The output of the neuron is represented by \( y_{(k)} \), where it is dependent on the activation
function $u_1(C_1)$ and the bias $b(k)$. There are several types of activation functions that were used in modeling neurons, some of them are fixed limiter function, linear function, sigmoid function, and bipolar sigmoid function as shown in Fig. 2.3 (Duch and Jankowski 1999; Dorofki et al. 2012).

### 2.1.2 Architecture

Connections between neurons with other neurons will form a layer pattern, so-called net architecture. Normally, ANN architecture consists of three different layers. The first layer is called the input layer. This layer acts as a receiver of data or input from the external stimuli. Incoming data is then sent to the next layer. In this
layer, the number of neurons can be more than one. There are no binding rules for determining the number of neurons; it depends on the number of entries to be used in the network. The next layer is a hidden layer. This layer contains neurons that can receive data or electrical signal than the previous layer of the input layer. Data or electrical signal that goes into these layers is processed using the functions available such as arithmetic, mathematics, etc. The hidden layer can contain one or more neurons, which depends on the suitability and complexity of the case at hand. Data processing results of this layer is then routed to the output layer. Output layer plays a role in determining the validity of data that are analyzed based on the existing limits in the activation function. The output of this layer can be used as a determinant of the outcome of the case at hand.

Based on the pattern of connections between neurons in the ANN, ANN architecture is divided into two types such as feedforward neural network and feedback neural network (Jain et al. 1996; Tang et al. 2007; Haykin 2009). Figure 2.4 shows the taxonomy of both the ANN architectures. Feedforward neural network is an ANN that does not have a feedback link on architecture. Data or incoming signals are allowed only to move in one direction only. This means that the output of each layer will not give any effect to the previous layer. In the architecture, it can be developed using a single layer or multiple layers. Usually, the multilayer component consists of three layers, namely a layer of input, output, and hidden. In a multilayer, hidden layer component plays a role in increasing the ability of computing power. One-layer perceptron, multilayer perceptron, and radial basis function are types of ANNs using feedforward neural networks.

Another architecture is a feedback neural network or repetitive. It has a design similar to the architecture of feedforward neural networks. However, in an architectural design there are additional feedbacks slow or feedback on the previous layer. This means the data or electrical signals that are allowed to propagate forward and feedback can be an input to the neurons before. This network is used for dynamic

![Fig. 2.4 Taxonomy of neural network architecture of feedforward and feedback neural networks adopted from Jain et al. (1996)](image-url)
applications such as adaptive control. Hopfield networks, Elman network, and Jordan network are some examples of the types of ANNs using feedback neural network.

2.1.3 Learning Process

ANN learning algorithm plays a role in the process of modifying the parameters and the value in the network to adapt its environment. The use of learning algorithms allows ANN assembles themselves for giving consistent response to input into the network. During the learning process, the parameters and the weights of synapses that are in the network will be modified. This is a form of response to the input stimulus to the output produced in accordance with the desired output. Level of learning will expire when the resulting output was consistent with the desired output.

To understand or design a learning process on ANN, there are three steps that need to be done by the designer (Jain et al. 1996). The steps are (1) learning paradigm, which refers to a process where a designer in building a system needs to choose the learning process in accordance with the information environment of the system; (2) learning algorithm, which refers to a learning rule that is used to modify the parameters and weights of synapses in the ANN series; and (3) finally, it is important to assess how much the network can be learned (capacity) and how many samples are required for training (sample complexity) as well as how fast the system can learn (time complexity).

Refers to the type of learning in the ANN, two types of learning processes have been widely adopted, namely supervised and unsupervised learning. Apparent differences between both are on the information provided by the network. Usually, the information given to supervised learning is in the form of sample patterns that have been marked or labeled, while in the unsupervised learning it occurs oppositely. Thus, for unsupervised learning it worked at random.

In supervised learning, a pattern that was given to the network has been known its output. Each incoming signals into a single neuron will continue to spread out along the network until the end layer of neurons in the output layer. In the final layer, the output pattern will be generated and then compared with the desired output pattern. Upon the occurrence of an error signal during the process of comparison between the output patterns generated by the pattern of the desired output, then the process should be modified to adjust the network weights so that the actual output will be in accordance with the desired output.

In contrast to supervised learning, unsupervised learning does not have guidelines or target output in the learning process. The network only receives many samples of an input and then puts the sample in any way into some classes or category. When the stimulus was given to the input layer, the response in the form of production category or class will have similar characteristics to the input stimulus. In contrast, the network will form a new coding, which led to a new class or category (Haykin 2009).
2.2 Adaptive Neuro-Fuzzy Interference System

Modify network-based fuzzy inference (ANFIS) is a combination of two soft-computing methods of ANN and fuzzy logic (Jang 1993). Fuzzy logic has the ability to change the qualitative aspects of human knowledge and insights into the process of precise quantitative analysis. However, it does not have a defined method that can be used as a guide in the process of transformation and human thought into rule base fuzzy inference system (FIS), and it also takes quite a long time to adjust the membership functions (MFs) (Jang 1993). Unlike ANN, it has a higher capability in the learning process to adapt to its environment. Therefore, the ANN can be used to automatically adjust the MFs and reduce the rate of errors in the determination of rules in fuzzy logic. This section will describe in details of the architecture of ANFIS, FISs, and network flexibility, and hybrid learning algorithm.

2.2.1 Fuzzy Inference System

A FIS was built on the three main components, namely basic rules, where it consists of the selection of fuzzy logic rules “If-Then;” as a function of the fuzzy set membership; and reasoning fuzzy inference techniques from basic rules to get the output. Figure 2.5 shows the detailed structure of the FIS. FIS will work when the input that contains the actual value is converted into fuzzy values using the fuzzification process through its membership function, where the fuzzy value has a range between 0 and 1. The basic rules and databases are referred to as the knowledge base, where both are key elements in decision-making. Normally, the database contains definitions such as information on fuzzy sets parameter with a function that has been defined for every existing linguistic variable. The development of a database typically includes defining a universe, determination of the number of linguistic values to

![Data Flow Diagram for Fuzzy Inference System](image_url)
be used for each linguistic variable, as well as establish a membership function. Based on the rules, it contains fuzzy logic operators and a conditional statement “If-Then.” The basic rules can be constructed either from a human or automatic generation, where the searching rules using input–output data numerically. There are several types of FIS, namely Takagi–Sugeno, Mamdani, and Tsukamoto (Cheng et al. 2005). A FIS of Takagi–Sugeno model was found to be widely used in the application of ANFIS method.

### 2.2.2 Adaptive Network

Adaptive network is one example of feedforward neural network with multiple layers (see Fig. 2.6). In the learning process, these networks often use supervised learning algorithm. In addition, adaptive network has the architecture characteristics that consists of a number of adaptive nodes interconnected directly without any weight value between them. Each node in this network has different functions and tasks, and the output depends on the incoming signals and parameters that are available in the node. A learning rule that was used can affect the parameters in the node and it can reduce the occurrence of errors at the output of the adaptive network (Jang 1993).

In learning the basic adaptive network, it is normally using gradient descent or back propagation and the chain rule. All this learning algorithms had been proposed by Werbos in 1970 (Jang 1993). Till date, gradient descent or back propagation is still used as a learning algorithm in an adaptive network. Even so, there are still found weaknesses in the backpropagation algorithm and further can reduce the capacity and accuracy of adaptive networks in making decisions. The slow convergence rate and tend to always stuck in local minima are major problems on backpropagation algorithm. Therefore, Jang (1993) have proposed an alternative learning algorithm, namely hybrid learning algorithm, which has the better ability to accelerate convergence and avoid the occurrence of trapped in local minima.

![Adaptive network](image)
2.2.3 ANFIS Architecture

ANFIS architecture is an adaptive network that uses supervised learning on learning algorithm, which has a function similar to the model of Takagi–Sugeno fuzzy inference system. Figure 2.7a, b shows the scheme fuzzy reasoning mechanism for Takagi–Sugeno model and ANFIS architecture. For simplicity, assume that there are two inputs \( x \) and \( y \), and one output \( f \). Two rules were used in the method of “If-Then” for Takagi–Sugeno model, as follows:

Rule 1 = If \( x \) is \( A_1 \) and \( y \) is \( B_1 \) Then \( f_1 = p_1x + q_1x + r_1 \)

Rule 2 = If \( x \) is \( A_2 \) and \( y \) is \( B_2 \) Then \( f_2 = p_2y + q_2y + r_2 \)

where \( A_1, A_2 \) and \( B_1, B_2 \) are the membership functions of each input \( x \) and \( y \) (part of the premises), while \( p_1, q_1, r_1 \) and \( p_2, q_2, r_2 \) are linear parameters in part-Then (consequent part) of Takagi–Sugeno fuzzy inference model.

Referring to Fig. 2.7, ANFIS architecture has five layers. The first and fourth layers contain an adaptive node, while the other layers contain a fixed node. A brief description of each layer is as follows:

Layer 1: Every node in this layer adapts to a function parameter. The output from each node is a degree of membership value that is given by the input of the

Fig. 2.7 a Sugeno fuzzy interference system “If-Then” and fuzzy logic mechanism. b ANFIS architecture (Suparta and Alhasa 2013)
membership functions. For example, the membership function can be a Gaussian membership function (Eq. 2.2), a generalized bell membership function (Eq. 2.3), or another type of membership function.

\[ \mu_{A_i}(x) = \exp \left[ -\left( \frac{x - c_i}{2a_i} \right)^2 \right] \]  
\[ \mu_{B_i}(x) = \frac{1}{1 + \left( \frac{x - c_i}{a_i} \right)^{2b}} \]  
\[ O_{1,i} = \mu_{A_i}(x), \quad i = 1, 2 \]  
\[ O_{1,i} = \mu_{B_{i-2}}(y), \quad i = 3, 4 \]

where \( \mu_{A_i} \) and \( \mu_{B_{i-2}} \) are the degree of membership functions for the fuzzy sets \( A_i \) and \( B_i \), respectively, and \( \{a_i, b_i, c_i\} \) are the parameters of a membership function that can change the shape of the membership function. The parameters in this layer are typically referred to as the premise parameters.

**Layer 2**: Every node in this layer is fixed or nonadaptive, and the circle node is labeled as \( \Pi \). The output node is the result of multiplying of signal coming into the node and delivered to the next node. Each node in this layer represents the firing strength for each rule. In the second layer, the T-norm operator with general performance, such as the AND, is applied to obtain the output

\[ O_{2i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \]

where \( w_i \) is the output that represents the firing strength of each rule.

**Layer 3**: Every node in this layer is fixed or nonadaptive and the circle node is labeled as \( N \). Each node is a calculation of the ratio between the \( i \)-th rules’ firing strength and the sum of all rules’ firing strengths. This result is known as the normalized firing strength.

\[ O_{3i} = \bar{w}_i = \frac{w_i}{\sum_i w_i} \]

**Layer 4**: Every node in this layer is an adaptive node to an output, with a node function defined as

\[ O_{4i} = \bar{w}_i f_i = \bar{w}_i (p_ix + q_iy + r_i) \]

where \( \bar{w}_i \) is the normalized firing strength from the previous layer (third layer) and \( (p_ix + q_iy + r_i) \) is a parameter in the node. The parameters in this layer are referred to as consequent parameters.
Layer 5: The single node in this layer is a fixed or nonadaptive node that computes the overall output as the summation of all incoming signals from the previous node. In this layer, a circle node is labeled as $\sum$.

$$O_{5i} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (2.9)$$

2.2.4 Hybrid Learning Algorithm

In the ANFIS architecture, the first layer and the fourth layer contain the parameters that can be modified over time. In the first layer, it contains a nonlinear of the premises parameter while the fourth layer contains linear consequent parameters. To update both of these parameters required a learning method that can train both of these parameters and to adapt to its environment. A hybrid algorithm proposed by Jang (1993) will be used in this study to train of these parameters. The use of this algorithm is due to the backpropagation algorithm that was used to train the parameters that exist in the adaptive networks found problematic especially in a slow convergence rate and tend to be trapped in local minima.

There are two parts of a hybrid learning algorithm, namely the forward path and backward path. In the course of the forward path, the parameters of the premises in the first layer must be in a steady state. A recursive least square estimator (RLSE) method was applied to repair the consequent parameter in the fourth layer. As the consequent parameters are linear, then RSLE method can be applied to accelerate the convergence rate in hybrid learning process. Next, after the consequent parameters are obtained, input data is passed back to the adaptive network input, and the output generated will be compared with the actual output.

While backward path is run, the consequent parameters must be in a steady state. The error occurred during the comparison between the output generated with the actual output is propagated back to the first layer. At the same time, parameter premises in the first layer are updated using learning methods of gradient descent or back propagation. With the use of hybrid learning algorithm that combines RSLE and the gradient descent methods, it can ensure the convergence rate is faster because it can reduce the dimensional search space in the original method of backpropagation (Nayak et al. 2004). One level of hybrid learning is called epochs. Table 2.1 describes briefly a hybrid learning process in ANFIS.

<table>
<thead>
<tr>
<th>Type</th>
<th>Path forwards</th>
<th>Path backwards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Premise parameter</td>
<td>Fixed</td>
<td>Gradient descent</td>
</tr>
<tr>
<td>Consequent parameter</td>
<td>RSLE</td>
<td>Fixed</td>
</tr>
<tr>
<td>Signal</td>
<td>Node output</td>
<td>Error rate</td>
</tr>
</tbody>
</table>

Table 2.1 Hybrid learning process
2.2.4.1 BackPropagation Learning for Parameter Premises

The premise parameters \( \{a, b, c\} \) in Eqs. 2.3 and 2.4 are adaptive parameters that can be trained to get the parameters in accordance with its environments. Suppose to have an adaptive network and similar to the Fig. 2.7b, where the network consists of five layers and has a total of \( N(L) \) node in layer-\( L \), then the number of square error in the \( L \) layer to \( p \) data is \( 1 \leq p \leq N \), and it can be defined as follows (Jang 1993; Jang and Sun 1995):

\[
E_p = \sum_{k=1}^{N(L)} d_k - X^L_{k,p}
\]  

(2.10)

where \( d_k \) is the \( k \)-th component of the vector of the desired output, while \( X^L_{k,p} \) is \( k \)-th component of the vector of actual output generated by adaptive network with input from the input vector \( p \). The main goal of adaptive learning system is to reduce errors that occur in the Eq. 2.10.

An early stage of learning begins by calculating the error rate of the output \( i \)-th node and \( L \) layer, with derivation equation as follows:

\[
\varepsilon_{L,i} = \frac{\partial E_p}{\partial L_i} = -2 \left( d_{i,p} - X^L_{i,p} \right)
\]  

(2.11)

For internal nodes in the \( l \) layer at \( i \) position, the error rate can be calculated using the Chain Rule

\[
\frac{\partial E_p}{\partial X^l_{i,i}} = \sum_{m=1}^{N(l+1)} \frac{\partial E_p}{\partial X^{l+1}_{m,p}} \frac{\partial X^{l+1}_{m,p}}{\partial X^l_{i,i}}
\]  

(2.12)

with \( 0 \leq l \leq L - 1 \). Internal node error signal can be expressed as a linear combination of the error rate in the layer node \( l (l + 1) \). Equation 2.12 is used to calculate the error signal at \( i \)-th layer node to \( l (l < L) \), while the use of Eq. 2.12 to reach the final layer. Further, when \( \alpha \) is a parameter used in some node, and then the equation will be obtained as follows:

\[
\frac{\partial E_p}{\partial \alpha} = \sum_{x^s \in S} \frac{\partial E_p}{\partial x^s} \frac{\partial x^s}{\partial \alpha}
\]  

(2.13)

where \( S \) is the set of nodes containing the parameter \( \alpha \), so that the whole issue of measurement error of \( \alpha \) will produce Eq. (2.14)

\[
\frac{\partial E}{\partial \alpha} = \sum_{p=1}^{P} \frac{\partial E_p}{\partial \alpha}
\]  

(2.14)
with steepest gradient descent method, the equation for repairing parameter $\alpha$ is obtained:

$$\Delta \alpha = -\eta \frac{\partial E}{\partial \alpha} \quad (2.15)$$

with $\eta$ is the learning rate process and stated as follows:

$$\eta = \frac{k}{\sqrt{\sum \alpha (\frac{\partial E}{\partial \alpha})^2}} \quad (2.16)$$

and $k$ is the step size, which can be changed in order to accelerate the convergence rate in adaptive networks.

### 2.2.4.2 Learning to Parameter Consequent RSLE

During the premises parameter in a steady state, then all output derived from the consequent parameters can be specified in a combination linear equation (Jang 1993; Jang and Sun 1995):

$$f = \bar{\omega}_1 f_1 + \bar{\omega}_2 f_2$$

$$= \bar{\omega}_1 (p_1 x + q_1 y + r_1) + \bar{\omega}_2 (p_2 x + q_2 y + r_2)$$

$$= (\bar{\omega}_1 x)p_1 + (\bar{\omega}_1 y)q_1 + (\bar{\omega}_1) r_1 + (\bar{\omega}_2 x)p_2 + (\bar{\omega}_2 y)q_2 + (\bar{\omega}_2) r_2 \quad (2.17)$$

When $N$ training data are given to Eq. 2.17, then the equation will be obtained as follows:

$$(\bar{\omega}_1 x)_1 p_1 + (\bar{\omega}_1 y)_1 q_1 + (\bar{\omega}_1)_1 r_1 + (\bar{\omega}_2 x)_2 p_2 + (\bar{\omega}_2 y)_2 q_2 + (\bar{\omega}_2)_2 r_2 = f_1$$

$$\vdots$$

$$(\bar{\omega}_1 x)_n p_1 + (\bar{\omega}_1 y)_n q_1 + (\bar{\omega}_1)_n r_1 + (\bar{\omega}_2 x)_n p_2 + (\bar{\omega}_2 y)_n q_2 + (\bar{\omega}_2)_n r_2 = f_n$$

(2.18)

To simplify, Eq. 2.18 can be expressed in matrix form as shown in Eq. 2.19:

$$A \theta = y \quad (2.19)$$

where $\theta$ is the vector $M \times 1$. $M$ refers to the number of elements that are consequent parameter set. While $A$ is the vector $P \times M$, where $P$ is the number of $N$ data training provided to the adaptive network and $y$ is the output vector $P \times 1$ whose elements are $N$ number of output data of an adaptive network. Normally, the amount of training data is larger than the number of consequent parameters, so the best
solution for $\theta$ is minimizing the squared error $\|A\theta = y^2\|$. By the least squares estimator (LSE), the equation for $\theta$ is defined as

$$
\theta^* = (A^T A)^{-1} A^T y
$$

(2.20)

where $A^T$ is the inverse of $A$ and if not singular, $(A^T A)^{-1}$ is the pseudo-inverse of $A$. By using a recursive LSE method, then the Eq. 2.20 becomes

$$
\begin{align*}
\theta_{i+1} &= \theta_i + P_{i+1} a_{i+1} (y_{i+1}^T - a_{i+1}^T \theta_i) \\
P_{i+1} &= P_i - \frac{P_i + a_i a_i^T P_i}{1 + a_i a_i^T} P_i a_i, \quad i = 0, 1, \ldots, P - 1
\end{align*}
$$

(2.21)

where $a_i^T$ is a row vector of the matrix $A$ in Eq. 2.19, $y_i$ is $i$-th element of $y$. $P_i$ sometimes called a covariance matrix and is defined by the following equation:

$$
P_i = (A^T A)^{-1}
$$

(2.22)

### 2.3 Linear Regression

In general, regression is a statistical method that can provide information about the patterns of relationships between two or more variables. In the regression method, it is identified two types of variables, namely (1) response variable or known also as the dependent variable; this variable is affected by other variables and usually denoted by $Y$, and (2) predictor variables are also known as independent variables, which are variables that are not affected by other variables and are usually denoted by $X$ (Shafiuallah et al. 2010).

The main goal in the regression analysis is to create a mathematical model that can be applied to forecast the values of the dependent variable based on the values of any variables. In use, the regression analysis is divided into two simple linear and multiple linear regressions. A simple regression analysis is a relationship between two variables, which are independent, and the dependent variables. In the multiple linear regression analysis, the relationship is found between three or more variables, which contain at least two independent variables and one dependent variable.

In the multiple linear regressions, the form of equation containing two or more variables is written as follows:

$$
Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_m X_m, \quad m = 1, 2, 3, \ldots, n
$$

(2.23)

where $\beta_0$ is a cutoff and $\beta_1 \ldots \beta_m$ are the regression coefficients. To obtain the values of the intercept and the regression coefficient in Eq. 2.23, the least squares method is frequently used (Brown 2009). Further, the use of such methods will be described in detail in Chap. 4 to develop an estimation model for ZPD.
References

McCulloch WS, Pitts WH (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5:115–133
Modeling of Tropospheric Delays Using ANFIS
Suparta, W.; Alhaza, K.M.
2016, XVI, 109 p. 51 illus., 23 illus. in color., Softcover
ISBN: 978-3-319-28435-4