

A Study of Neuro-fuzzy System in Approximation-based Problems

¹Lim Eng Aik & ²Yogan S/O Jayakumar

¹Institute of Engineering Mathematics, University Malaysia Perlis, Jejawi, 02600 Arau, Perlis, Malaysia

²Faculty of Engineering, Multimedia University, Jalan Multimedia, 63100 Cyberjaya, Selangor, Malaysia
e-mail: ¹ealim@unimap.edu.my

Abstract The Fusion of Artificial Neural Networks (ANN) and Fuzzy Inference System (FIS) has attracted a growing interest of researchers in various scientific and engineering areas due to the growing need for adaptive intelligent systems to solve real world problems. ANN learns by adjusting the interconnections between layers. FIS is a popular computing framework based on the concept of fuzzy set theory and fuzzy if-then rules. The advantages of the combination of ANN and FIS are apparent. This paper implements a hybrid neuro-fuzzy system underlying ANFIS (Adaptive Neuro-Fuzzy Inference System), a fuzzy inference system implemented in the framework of neural networks. The motivation stems from a desire to achieve performance in terms of accuracy and several simulations studies regarding the determination of the optimal number of membership functions have been done. In our simulations, we utilize the ANFIS architecture to model nonlinear functions. In addition, the effects of using different types of membership functions were compared. Based upon numerical evidence, some general guidelines for choosing the number of membership function have been proposed. To experiment with the technique that allows the combination of neural network and fuzzy system, we have implemented ANFIS to a real world application (Phytoplankton concentration problem); and yielding good results.

Keywords ANFIS; Neuro-fuzzy system; membership function; function approximation.

1 Introduction

System modeling based on conventional mathematical tools (e.g., differential equations) is not well suited for dealing with ill-defined and uncertain systems. By contrast, a fuzzy inference system employing fuzzy *if then* rules can model the qualitative aspects of human knowledge and reasoning processes without employing precise quantitative analyses. This *fuzzy modeling* or *fuzzy identification*, first explored systematically by Takagi & Sugeno [17], has found numerous practical applications in control (Sugeno [14], Pedrycz [11]), prediction and inference (Kandel [4], Kandel [5]). However, there are some basic aspects of this approach which are in need of better understanding. More specifically:

There is a need for effective methods for tuning the number of membership functions (MF's) so as to minimize the output error measure or maximize performance index (Jang [1]).

In this perspective, the aim of this paper is to study and suggest some general guidelines for selecting the number of membership function for Adaptive-Network-based Fuzzy Inference System or simply ANFIS (Jang, 1993) that would minimize the output error measure for one to three dimensional approximation-based problems. The next section gives the review on Neuro-Fuzzy System. Section 3 introduced the basics of ANFIS. Section 4 explains how to proceed membership function selection for ANFIS. Simulation results for nonlinear function and Phytoplankton concentration problem are given in section 5. Section 6 concludes this paper by giving important extensions and future directions of this work.

2 Neuro-Fuzzy System

Neural networks and fuzzy systems both are stand-alone systems. With the increase in the complexity of the process being modeled, the difficulty in developing dependable fuzzy rules and membership functions increases. This has led to the development of another approach which is mostly known as neuro-fuzzy approach. It has the benefits of both neural networks and fuzzy logic and is attracting an army of researchers in this field. Defining the structure and size of neural networks and determining fuzzy rules and the membership functions systematically are main research areas concerning this AI technique (Lin [7]). The neuro-fuzzy hybrid system combines the advantages of fuzzy logic system, which deal with explicit knowledge that can be explained and understood, and neural networks, which deals with implicit knowledge.

One of the advantages of fuzzy systems is that they describe fuzzy rules, which fit the description of real-world processes to a greater extent. Another advantage of fuzzy systems is their interpretability; it means that it is possible to explain why a particular value appeared at the output of a fuzzy system. In turn, some of the main disadvantages of fuzzy systems are that expert's knowledge or instructions are needed in order to define fuzzy rules, and that the process of tuning of the parameters of the fuzzy system (e.g. parameters of the membership functions) often requires a relatively long time. Both these disadvantages are related to the fact that it is not possible to train fuzzy systems.

A diametrically opposite situation can be observed in the field of neural networks. You can train neural networks, but it is extremely difficult to use a prior knowledge about the considered system and it is almost impossible to explain the behavior of the neural network system in a particular situation.

In order to compensate the disadvantages of one system with the advantages of another system, several researchers tried to combine fuzzy systems with neural networks. A hybrid system named *ANFIS (Adaptive-Network-Based Fuzzy Inference System)* has been proposed by Jang [1]. Fuzzy inference in this system is realized with the aid of a training algorithm, which enables to tune the parameters of the fuzzy system.

3 ANFIS: Adaptive Neuro-Fuzzy Inference System

This section introduces the basics of ANFIS network architecture and its hybrid learning rule. A detailed coverage of ANFIS can be found in (Jang et al. [2], Jang [1]). The Sugeno fuzzy model was proposed by Sugeno et al. [15], Takagi & Sugeno [17] in an effort to formalize a systematic approach to generating fuzzy rules from an input-output data set. A typical fuzzy rule in a Sugeno fuzzy model has the format

If x is A and y is B then $z = f(x,y)$,

where A and B are fuzzy sets in the antecedent; $z = f(x,y)$ is a crisp function in the consequent.

Usually $f(x,y)$ is a polynomial in the input variables x and y , but it can be any other functions that can appropriately describe the output of the system within the fuzzy region specified by the antecedent of the rule. When $f(x,y)$ is a first-order polynomial, we have the first-order Sugeno fuzzy model, which was originally proposed in Sugeno et al. [15], Takagi & Sugeno [17]. When f is a constant, we then have the zero-order Sugeno fuzzy model, which can be viewed either as a special case of the Mamdani fuzzy inference system (Mamdani et al. [8]) where each rule's consequent is specified by a fuzzy singleton, or a special case of Tsukamoto's fuzzy model (Tsukamoto, [18]) where each rule's consequent is specified by a membership function of a step function centered at the constant. Moreover, a zero order Sugeno fuzzy model is functionally equivalent to a radial basis function network under certain minor constraints (Jang et al. [3]).

Consider a first-order Sugeno fuzzy inference system which contains two rules:

Rule 1: If X is A_1 and Y is B_1 , then
 $f_1 = p_1x + q_1y + r_1$,

Rule 2: If X is A_2 and Y is B_2 , then
 $f_2 = p_2x + q_2y + r_2$.

Figure 1(a) illustrates graphically the fuzzy reasoning mechanism to derive an output f from a given input vector $[x, y]$. The firing strengths w_1 and w_2 are usually obtained as the product of the membership grades in the premise part, and the output f is the weighted average of each rule's output.

To facilitate the learning of the Sugeno fuzzy model, it is convenient to put the fuzzy model into framework of adaptive networks that can compute gradient vectors systematically. The resultant network architecture, called ANFIS, is shown in Figure 1(b), where node within the same layer performs functions of the same type, as detailed below. (Note that O_i^j denotes the output of the i -th node in j -th layer.)

Layer 1 Each node in this layer generates a membership grades of a linguistic label. For instance, the node function of the i -th node may be a generalized bell membership function:

$$O_i^1 = \mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i} \right)^2 \right]^{b_i}} \quad (1)$$

where x is the input to node i ; A_i is the linguistic label (small, large, etc.) associated with this node; and $\{a_i, b_i, c_i\}$ is the parameter set that changes the shapes of the membership function. Parameters in this layer are referred to as the premise parameters.

Layer 2 Each node in this layer calculates the firing strength of a rule via multiplication:

$$O_i^2 = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2. \quad (2)$$

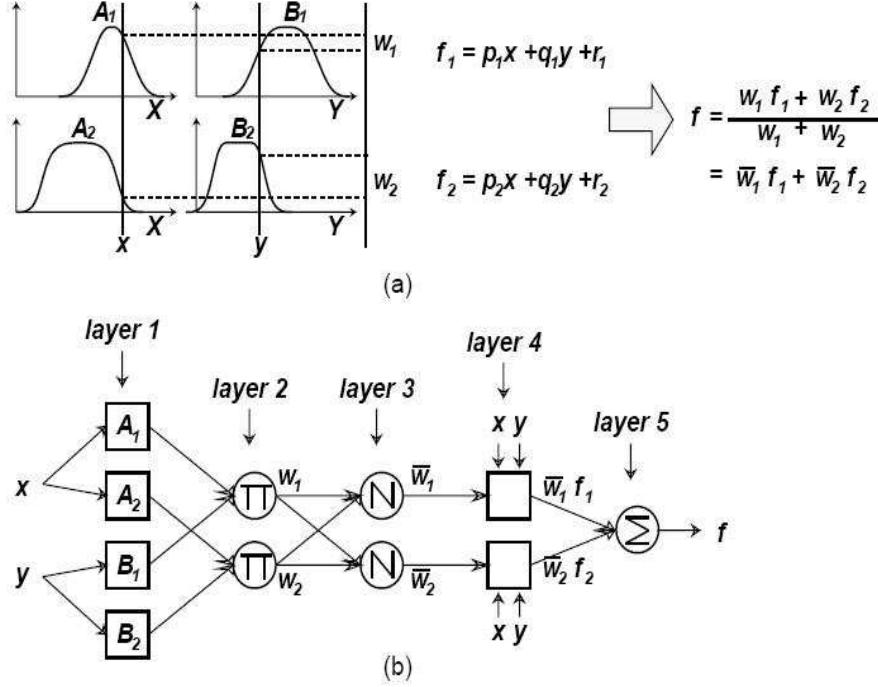


Figure 1: (a) First-order Sugeno Model; (b) Corresponding ANFIS Architecture

Layer 3 Node i in this layer calculates the ratio of the i -th rule's firing strength to total of all firing strengths:

$$O_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2. \quad (3)$$

Layer 4 Node i in this layer compute the contribution of i -th rule toward the overall output, with the following node function:

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i), \quad (4)$$

where $\bar{w}_i$ is the output of layer 3, and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer will be referred to as the consequent parameters.

Layer 5 The single node in this layer computes the overall output as the summation of contribution from each rule:

$$O_i^5 = f = \sum_i \bar{w}_i f_i. \quad (5)$$

The constructed adaptive network in Figure 1(b) is functionally equivalent to a fuzzy inference system in Figure 1(a). The basic learning rule of ANFIS is the backpropagation gradient descent (Werbos [20]), which calculates error signals (the derivative of the squared error with respect to each node's output) recursively from the output layer backward to

the input nodes. This learning rule is exactly the same as the backpropagation learning rule used in the common feedforward neural networks (Rumelhart et al. [13]). From the ANFIS architecture in Figure 1, it is observed that given the values of premise parameters, the overall output f can be expressed as linear combinations of the consequent parameters:

$$\begin{aligned} f &= \bar{w}_1 f_1 + \bar{w}_2 f_2 \\ &= \bar{w}_1(p_1 x + q_1 y + r_1) + \bar{w}_2(p_2 x + q_2 y + r_2) \end{aligned} \quad (6)$$

3.1 ANFIS Learning Method

In conventional neural networks, the backpropagation algorithm is used to learn, or adjust weights on connecting arrows between neurons from input-output training samples. In the ANFIS structure, the parameters of the premises and consequents play the role of weights. Specifically, the shape of membership functions in the “If” part of the rules is determined by a finite number of parameters. These parameters are called premise parameters, whereas the parameters in the “Then” part of the rules are referred to as consequent parameters. The ANFIS learning algorithm (Jang [1]) consists of adjusting the above set of parameters.

For ANFIS, a mixture of backpropagation and least square estimation (LSE) is used. Backpropagation is used to learn the premise parameters, and LSE is used to determine the parameters in the rules’ consequents. A step in the learning procedure has two passes. In the forward pass, node outputs go forward, and the consequent parameters $\{p_i, q_i, r_i\}$ are estimated by least squares method, while the premise parameters remain fixed. In the backward pass the error signals are propagated backwards, and backpropagation is used to modify the premise parameters $\{a_i, b_i, c_i\}$, while consequent parameters remain fixed. This combination of least-squares and backpropagation methods are used for training FIS membership function parameters to model a given set of input/output data. The performance of this system will be evaluated using RMSE, root mean square errors (difference between the FIS output and the training/testing data output), defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{k=1}^n (y_k - o_k)^2} \quad (7)$$

where y_k is the desired output and o_k is the actual system output. n is the number of training/testing samples. We will use this as our measure of error to obtain training and testing error later in our simulation studies.

4 Membership Function Selection for ANFIS

Every practitioner of neuro-fuzzy system faces the same architecture selection problem: How many membership functions to choose for each input? In a conventional fuzzy inference system, the number of rules is decided by an expert who is familiar with the system to be modeled. In our simulation, however, no expert is available and the number of membership functions (MF’s) assigned to each input variable is chosen empirically, i.e., by examining the desired input-output data and/or by trial and error method (we tried with different number of membership function in range [1, 25] for each dimensional nonlinear function study and we only show the number of membership function that gives the minimal output error in

table with respect to type of membership function for each study). This situation is much the same as that of neural networks; there are no simple ways to determine in advance the minimal number of hidden nodes necessary to achieve a desired performance level.

After the number of MF's associated with each inputs are fixed, the initial values of premise parameters are set in such a way that the MF's are equally spaced along the operating range of each input variable. Moreover, they satisfy ε -completeness (Lee [6]) with $\varepsilon = 0.5$, which means that given a value x of one of the inputs in the operating range, we can always find a linguistic label A such that $\mu_A(x) \geq \varepsilon$. In this manner, the fuzzy inference system can provide smooth transition and sufficient overlapping from one linguistic label to another. Though we did not attempt to keep the ε -completeness during the learning in our simulation, it can be easily achieved by using the constrained gradient method (Wismer et al. [21]). Figure 2 shows a typical initial MF setting when the number of MF is 3 and the operating range is $[0, 0.5]$. Note that throughout the simulation examples presented below, all the membership functions used are the **bell function** defined in equation:

$$\mu_A(x) = \frac{1}{1 + \left[\left(\frac{x-c}{a} \right)^2 \right]^b} \quad (8)$$

which contains three fitting parameters a , b and c . Each of these parameters has a physical meaning: c determines the center of the corresponding membership function; a is a half width; and b (together with a) controls the slope at the crossover points (where MF value is 0.5). Figure 3 shows these concepts.

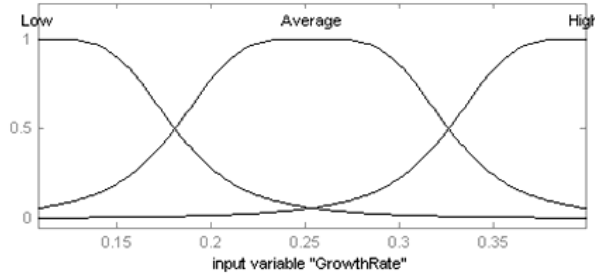


Figure 2: A typical initial membership function setting in our simulation. (The operating range is assumed to be $[0, 0.5]$)

In the next section, several simulation studies regarding the selection of number of membership functions have been done. Also, the effect of different types of membership functions was compared: triangular, trapezoidal, generalized bell shape and Gaussian functions. AN-FIS model was used since it is a well-known neuro-fuzzy system for function approximation (Jang [1]). The data based approaches are by evaluating functions from training and testing data. At the end of that section, some general guidelines for selecting the number of membership functions are proposed.

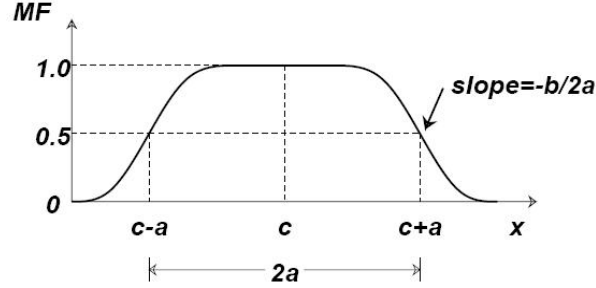


Figure 3: Physical meanings of the parameters in bell membership function

5 Simulation Results

This section presents the simulation results of one to three dimensional nonlinear functions. In this and the following sections, we search the optimal number of membership function for approximating nonlinear functions. Also, the effect of different types of membership functions was compared: triangular, trapezoidal, generalized bell shape and Gaussian functions. To avoid biasness, we work on three different functions for each section.

5.1 One-Dimensional Nonlinear Functions

Simulation One: Let us consider a 1-D example, a function described by

$$y = x^3 + 0.3x^2 - 0.4x, \quad x \in [-1, 1]. \quad (9)$$

The training set composes of 100 points, which are chosen by uniformly partitioning the domain $[-1, 1]$ with grid size of 0.02.

Simulation Two: In this example, we consider using ANFIS to model

$$y = \sin(x)/\cos(x) + \cos(x); \quad x \in [0, 1]. \quad (10)$$

The training set consists of 100 points, which are chosen by uniformly partitioning the domain $[0, 1]$ with grid size of 0.1.

Simulation Three: The target function is described by

$$y = x^4 + 0.2x - 1; \quad x \in [0, 1]. \quad (11)$$

The training set consists of 100 points, which are chosen by uniformly partitioning the domain $[0, 1]$ with grid size of 0.01. All the three simulation take the test set comprises 100 points randomly sampled from the same domain of each simulation case. Simulation results for Simulation One, Simulation Two and Simulation Three are shown in Table 1, Table 2 and Table 3.

Table 1: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	16	0.000331957	0.00010623
trapmf	21	0.000254998	0.00026045
gbellmf	17	0.000628877	0.00098509
gaussmf	18	0.000388659	0.00050377

Table 2: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	20	0.000193956	0.00020000
trapmf	20	0.000135695	0.00037968
gbellmf	17	0.000123685	0.00019113
gaussmf	14	9.1094e-005	0.00013005

5.2 Two-Dimensional Nonlinear Functions

Simulation Four: Let us consider a 2-D example, a function described by

$$z = \sin(10x) \sin(10y); \quad x, y \in [0, 1]. \quad (12)$$

The training set composes of 100 points, which are chosen by uniformly partitioning the domain $[0, 1]$ with grid size of 0.01.

Simulation Five: In this example, we consider using ANFIS to model

$$z = 0.5x^2 + 0.5y^2; \quad x, y \in [-2, 2]. \quad (13)$$

The training set consists of 100 points, which are chosen by uniformly partitioning the domain $[-2, 2]$ with grid size of 0.04.

Simulation Six: The target function is described by

$$z = (5/(2\pi)) \exp(-(x^2 + y^2)/2); \quad x, y \in [-4, 4]. \quad (14)$$

The training set consists of 100 points, which are chosen by uniformly partitioning the domain $[-4, 4]$ with grid size of 0.08. All the three simulations take the test set comprises 100 points randomly sampled from the same domain for each simulation case. Simulation results for Simulation Four, Simulation Five and Simulation Six are shown in Table 4, Table 5 and Table 6.

Table 3: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	13	0.000418980	0.00011745
trapmf	16	0.000637919	0.00116700
gbellmf	16	0.000407868	0.00048704
gaussmf	14	0.000227635	0.00026164

Table 4: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error	Testing error
trimf	6	0.0025774	0.079571
trapmf	5	0.0113076	0.166510
gbellmf	5	0.0061942	0.049722
gaussmf	5	0.0020153	0.021969

5.3 Three-Dimensional Nonlinear Functions

Simulation Seven: Let us consider a 3-D example, a function described by

$$y = \exp(x_1) + 2 \cos(x_2) + x_3^{1.5}; \quad x_1, x_2, x_3 \in [0, 1]. \quad (15)$$

Simulation Eight: In this example, we consider using ANFIS to model

$$y = x_1^2 + 2 \cos(x_2) + x_3^3; \quad x_1, x_2, x_3 \in [0, 1]. \quad (16)$$

Simulation Nine: The target function is described by

$$y = 2\pi \exp(-x_1^2 - x_2^2) + \sin(x_3); \quad x_1, x_2, x_3 \in [0, 1]. \quad (17)$$

Simulation Ten, Eleven and Twelve have the training set composes of 100 points, which are chosen by uniformly partitioning the domain $[0,1]$ with grid size of 0.01. And the test set comprises 100 points randomly sampled from the same domain. Simulation results for Simulation Seven, Simulation Eight and Simulation Nine are shown in Table 7, Table 8 and Table 9.

5.4 Discussion of the Simulation Results

In Simulation One, Two and Three, we observe that the optimal numbers of membership functions needed for best results is between 13 and 21. Refer Table 1, Table 2 and Table 3. Whereas in Simulation Four, Five and Six, the range is between 3 and 9. Refer Table 4, Table 5 and Table 6. While in Simulation Seven, Eight and Nine, we observe that the optimal number membership function needed is between 2 and 3. Refer Table 7, Table 8

Table 5: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error	Testing error
trimf	3	0.0100876	0.010405
trapmf	3	0.1016360	0.151700
gbellmf	5	0.0604684	0.086995
gaussmf	6	0.0355789	0.060988

Table 6: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	5	0.00423443	0.026675
trapmf	9	0.00447680	0.057362
gbellmf	5	0.00258930	0.039697
gaussmf	5	0.00234420	0.029570

and Table 9. Based from these results, we observe that there is a relationship between the number of membership function and the dimension of the function we intend to approximate. As the dimension of a function increases, the number of membership function required for optimal result reduces. This can be explained in terms of the number of parameters used in the model. The total number of parameters is:

$$n_{input} \cdot n_{mf} \cdot n_{mf_parameters} + n_{mf}^{n_{input}} \cdot (n_{input} + 1)$$

From the above formula, as the number of input increases, while the number of membership function is fixed, the total number of parameters will also increase. In this case, if the number of membership function is big, we will have too many parameters, which can result over-fitting and the model is not useable. Thus, it is reasonable to reduce the number of membership function when the number of inputs increases.

This discussion has motivated us to suggest some general guidelines for selecting the number of membership function. Let us recall the question arise in previous Section: *What is the optimal number of membership function for a given problem?* In a real world problem this is often a tiring process of trial and error (Nauck [10]).

Based from the simulation results, we break down the question regarding the type of “problem”. We can now ask the questions: (i) What is the optimal number of membership function for a 1-D problem? (ii) What is the optimal number of membership function for a 2-D problem? (iii) What is the optimal number of membership function for a 3-D problem? The following guidelines are proposed.

Guideline One: If we are given a one dimensional problem, we suggest that the choice of the initial number of membership function would be between 12 and 22. This initial choice is made based from the results from Simulation One, Two and Three.

Table 7: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	3	0.00445752	0.0066389
trapmf	2	0.02360820	0.0521910
gbellmf	2	0.01404680	0.0280830
gaussmf	2	0.00866257	0.0188540

Table 8: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	3	0.0120845	0.023696
trapmf	2	0.0381719	0.063377
gbellmf	2	0.0223064	0.036611
gaussmf	2	0.0137770	0.026364

Guideline Two: In the case of two dimensional problems, we suggest that the first trial for the number of membership function would be between 3 and 10. This suggestion is made based from the results in Simulation Four, Five and Six.

Guideline Three: From the results we get in three dimensional problems, we notice that selecting two as the initial number of membership function gives best result for most of the cases in a three dimensional problem. Observe the results in Table 7, Table 8 and Table 9. Thus, for this case it is appropriate to choose the number of membership function to be 2.

We would like to remind that the initial choice meant in our guideline is not the optimal number of membership function, but close to the optimal number of membership function needed for satisfactory performance.

Also, we have studied the characteristic of the type of membership function. In each simulation, we compared the effect of different types of membership function. Here, the triangular, trapezoidal, generalized bell shapes and Gaussian functions were used.

Let us consider for the case of two dimensional function approximations. We can clearly observe that in Simulation Five, triangular membership function is better than the others. See Table 5. For the same case, we notice that, in Simulation Four, Gaussian membership function gives better results. See Table 4. Thus, the choice of a membership function depends on a particular application involved.

In most of the simulations, we can observe that the effect of different types of membership function does not make vast difference in the output results compared to using different number of membership functions. As a result, as long as the number of membership function is adequate, the choice of the type of membership function is not critical.

Table 9: Training and Testing Error Versus Optimal Number of Membership Functions (Mf) and Different Types of Membership Function

	Number of Mf	Training error (RMSE)	Testing error (RMSE)
trimf	3	0.0257155	0.051697
trapmf	2	0.0477141	0.083205
gbellmf	2	0.0281959	0.048940
gaussmf	2	0.0166032	0.027928

5.5 Simulation of Real World Problem (Phytoplankton Concentration Problem)

We will consider a lake problem in this case study. Here we wish to estimate the concentration of phytoplankton in a lake (Thomann & Mueller [19]). The effects of both the growth rate and death rate of phytoplankton will be studied. This is an effort to produce appropriate control strategies such as physical, chemical, and biological treatment.

Consider the equation

$$P = l_o \exp((G_r - D_r)t); \quad (18)$$

as a representative of phytoplankton concentration for initial condition l_o . This equation is obtained by solving the differential equation (Thomann & Mueller, 1987).

$$\frac{dP}{dt} = (G_r - D_r)P \quad (19)$$

P = phytoplankton concentration

G_r = the growth rate of phytoplankton l_o = initial concentration

D_r = the death rate of phytoplankton t = time

Units for each term above are;

P : microgram per litre [$\mu g l^{-1}$] G_r : per day [$(day)^{-1}$]

l_o : microgram per litre [$\mu g l^{-1}$] D_r : per day [$(day)^{-1}$]

t : day

In this case study, the Adaptive Neuro-Fuzzy Inference System (ANFIS) is implemented. ANFIS modeling can be ecologically interpreted (Rafael [12]). Knowledge about the increase and decrease of phytoplankton concentration in water body can help water managers make better control decisions. In our study for this paper, we will estimate the phytoplankton concentration during earlier time, say day three. Let the initial concentration to be $15 \mu g l^{-1}$. Thus, the target function (19) can be written as;

$$P = 15 \exp((G_r - D_r)3), \quad G_r, D_r \in [0.1, 0.4]. \quad (20)$$

The training set consists of 100 pairs, which are chosen by evaluating 100 random pairs from the domain $[0.1, 0.4]$. And the testing set comprises 100 pairs randomly sampled from the same domain. Based on the guideline proposed in the previous chapter, the initial number of membership functions chosen for this two dimensional problem is 3. Here we use the generalized bell function. After training, we obtained $RMSE_{train} = 0.01077$

Table 10: Results of ANFIS Model with Different Number of Membership Functions

Number of Mf	Training error (RMSE)	Testing error (RMSE)
2	0.070710	0.078059
3	0.010770	0.026585
4	0.010944	0.046206
5	0.012877	0.934920
6	0.017265	1.599100

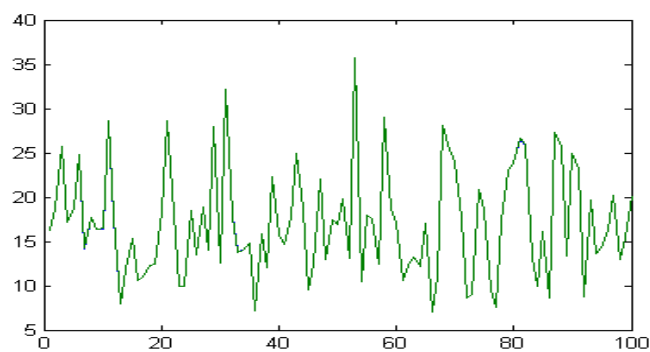


Figure 4: ANFIS Output Using Training Data

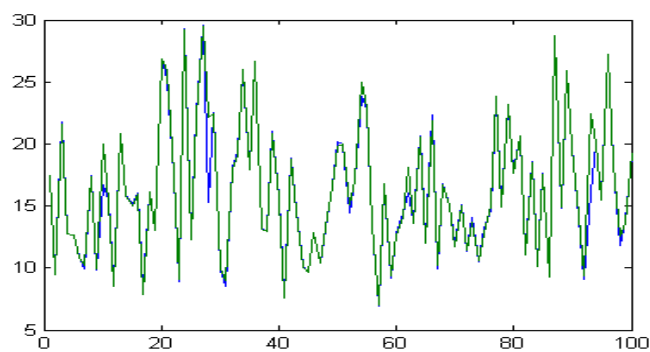


Figure 5: ANFIS Output Using Testing Data

and $RMSE_{test} = 0.026585$, as the best result when we employed a different number of membership functions (Table 10).

The knowledge describing the system's behavior is represented by the membership functions defining the linguistic variables. Two linguistic variables are defined; growth rate and death rate. Each variable is given three membership functions, with linguistic values: *Low*, *Average*, and *High*. Table 11 is a summary of the linguistic variables and linguistic values. Figure 6 illustrates the ANFIS structure developed for the estimation of phytoplankton concentration.

Table 11: Linguistic Variables and Linguistic Values for Fuzzy Rules

Linguistic Variable	Linguistic Values
Growth rate	{low, average, high}
Death rate	{low, average, high}

Figure 7 and Figure 8 show the initial and final (tuned) membership functions for growth rate and death rate. The changes in the shape of the membership functions are due to changes in values of the each membership function parameters.

The linguistic variables, growth rate and death rate with the linguistic values assigned in Table 11 are used to develop the Sugeno fuzzy rules for the estimation of phytoplankton concentration. Generally the number of rules for any system is the product of the number of linguistic values of all the linguistic variables. Since we have three linguistic values {low, average, high} for each input, we have a total of 9 fuzzy rules. The nine rules that describe the phytoplankton concentration estimation are summarized in Table 12.

Table 12: Sugeno Fuzzy Rules for Estimating Phytoplankton Concentration

Rule	IF Growth rate is	AND Death rate is	THEN Phytoplankton concentration
R^1	Low	low	$c_1 \bullet V$
R^2	low	average	$c_2 \bullet V$
R^3	low	high	$c_3 \bullet V$
R^4	average	low	$c_4 \bullet V$
R^5	average	average	$c_5 \bullet V$
R^6	average	high	$c_6 \bullet V$
R^7	high	low	$c_7 \bullet V$
R^8	high	average	$c_8 \bullet V$
R^9	high	high	$c_9 \bullet V$

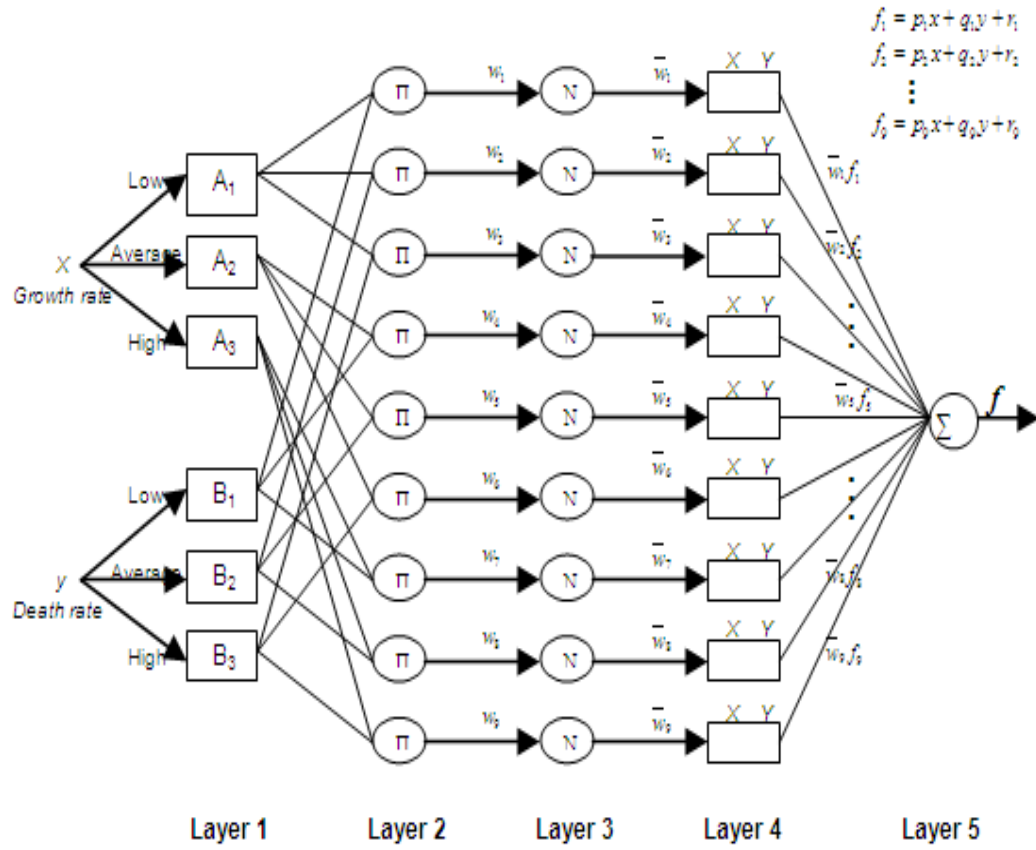


Figure 6: ANFIS structure for estimating phytoplankton concentration

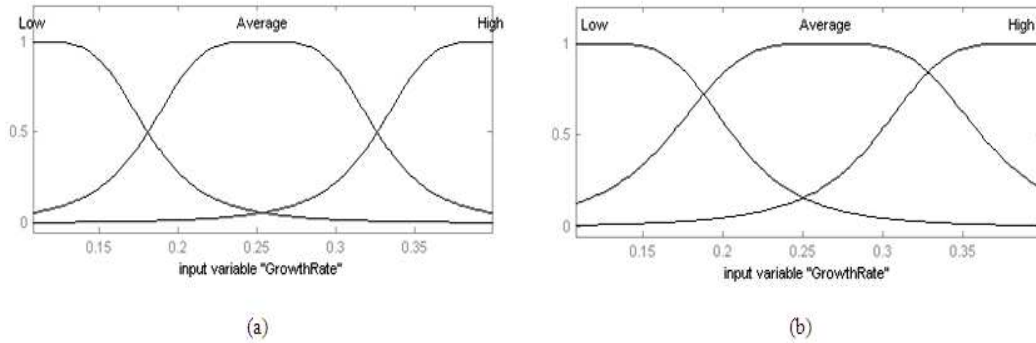


Figure 7: (a) Initial and (b) Final Membership Functions for Growth Rate

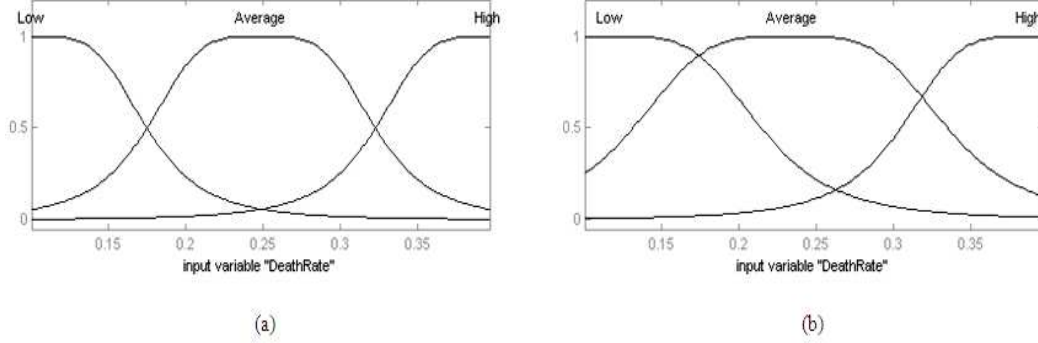


Figure 8: (a) Initial and (b) Final Membership Functions for Death Rate

In Figure 12, $V = [x, y, 1]^T$ and c_i is the i -th row of the following parameter matrix C :

$$C = \begin{bmatrix} 52.42 & -34.64 & 13.77 \\ 32.41 & -23.38 & 12.27 \\ 19.62 & -19.73 & 12.09 \\ 67.09 & -61.96 & 13.66 \\ 46.93 & -40.24 & 13.82 \\ 26.3 & -28.27 & 14.55 \\ 98.74 & -99.01 & 9.197 \\ 62.78 & -63.11 & 14.41 \\ 38.85 & -50.46 & 20.03 \end{bmatrix}.$$

These values are the final values of the consequent parameters p_i , q_i , and r_i , where c_i represents $\{p_i, q_i, r_i\}$. The training data is used to train the ANFIS and tune the premise and consequent parameters. The premise parameters are the parameters of the generalized bell function from equation (8). These parameters are updated in Layer 1 (see Figure 6). Figure 7 and Figure 8 show the initial and final (tuned) membership functions for growth rate and death rate respectively. The outputs of Layer 4 (see Figure 6) are calculated based on the consequents parameters.

Parameter matrix C illustrates the final values of these consequent parameters. Referring to Figure 4 and Figure 5, we observe that the model provides a good generalization; it worked well for different data.

From the results of this application, we conclude that the ANFIS produces satisfactory results in terms of its performance. Due to the ‘fuzzy’ nature of the variables, the ANFIS architecture is ideally suited for applications that require rule-base reasoning as part of the decision making process.

6 Conclusion

In this paper we have presented the modeling of a neuro-fuzzy system. The advantages of the combination of ANN and FIS are obvious. The performance of ANFIS (Adaptive Neuro-Fuzzy Inference System) using different number of membership functions was investigated

in this paper. The number of membership functions can determine the performance of a neuro-fuzzy system, in terms of reducing the size of error and generalization. In this paper, simulation evidence explains that we can make a choice regarding the problem of finding appropriate number of membership functions which is often a tiring process of trial and error.

The choice of the number of membership function which has been a benchmark problem is investigated. From the simulation results obtained, we notice that there are similarities in the range, where the optimal number of membership functions falls. Here in this paper, we have investigated three different dimensional problems; 1D, 2D and 3D. The numerical evidence shows that the choice of optimal membership function can now be done based upon the dimension of a particular problem.

We also tested ANFIS learning in the problem of estimating the concentration of phytoplankton in a lake. From the results obtained, ANFIS is shown to be capable of providing good performance for this function approximation-based application. In this problem, the selection of membership function is made using the guidelines suggested from the Modeling results.

For future work, finding theoretical proof for determining the optimal number of membership function would be a very interesting and challenging problem. Considering higher dimensional function for the search of optimal number of membership function would also be interesting.

Another comparison that we have done in this paper is the effect of using different type of membership functions. As we were able to make the choice of number of membership function for different dimensional problem, the selection of the type of membership function was not an easy task. Simulation results show that the type of membership function that gives better results varies for different problems. The question: Why different type of membership function gives different results, would be also interesting for future work.

References

- [1] J.S.R. Jang, *ANFIS: Adaptive-network-based Fuzzy Inference Systems*, IEEE Trans. on Systems, Man. and Cybernetics, vol. 23, no 3(1993), 665-685.
- [2] J.S.R. Jang et al, *Fuzzy Modelling Using Generalized Neural Networks and Kalman Filter Algorithm*, In proceedings of the Ninth National Conference on Artificial Intelligent (A A AI-91)(1991), 762-767.
- [3] J.S.R. Jang et al, *Neuro-fuzzy Modelling and Control*, The proceedings of the IEEE, 83(3)(1995), 378-406.
- [4] A. Kandel, *Fuzzy Expert Systems*, Addison-Wesley, 1988.
- [5] A. Kandel, *Fuzzy Expert Systems*, CRC Press, Boca Raton, Fl., 1992.
- [6] C.C. Lee, *Fuzzy Logic in Control Systems: Fuzzy Logic Controller*, IEEE Trans. On systems, Man and Cybernetics, 20(2)(1990), 404-435.
- [7] C.T. Lin, *Neural Fuzzy Control Systems with Structure and Parameter Learning*, World Scientific, Singapore, 1994.

- [8] E.H. Mamdani et al, *An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller*, International Journal of Man-Machine studies, 7(1)(1975), 1-13.
- [9] D. Nauck, & R. Kruse, *Designing Neuro-Fuzzy Systems Through Backpropagation, Fuzzy Modeling: Paradigms and Practice*, pp. 203-228, Kluwer, Boston, 1996.
- [10] D. Nauck, & R. Kruse, *Neuro-Fuzzy Systems for Function Approximation*, Fuzzy Sets and Systems 101(1999), 261-271
- [11] W. Pedrycz, *Fuzzy Control and Fuzzy Systems*, Wiley, New York, 1989.
- [12] M. Rafael, *A Neuro-fuzzy Modeling Tool to Estimate Fluvial Nutrient Loads in Watersheds Under Time-varying Human Impact*, American Society of Limnology and Oceanography, (2004), 342-355
- [13] D.E. Rumelhart et al, *Learning Internal Representations by Error Propagation; Parallel Distributed Processing: explorations in the microstructure of cognition*, volume 1, chapter 8, pages 318-362, MIT press, Cambridge, MA, 1986.
- [14] M. Sugeno(editor), *Industrial Applications of Fuzzy Control*, Elsevier Science Pub. Co., 1985.
- [15] M. Sugeno et al, *Structure Identification of Fuzzy Model*, Fuzzy Sets and Systems, 28(1988), 15-33.
- [16] H. Takagi, *Fusion Technology of Fuzzy Theory and Neural Networks – Survey and Future Directions*. In Proc. 1st Int. Conf. on Fuzzy Logic & Neural Networks, pp. 13-26, 1990.
- [17] T. Takagi, & M. Sugeno, *Fuzzy Identification of Systems and Its Applications to Modeling and Control*, IEEE Trans. on Systems, Man, and Cybernetics, 15(1985), 116-132.
- [18] Y. Tsukamoto, *An Approach to Fuzzy Reasoning Method; Advances in Fuzzy Set Theory and Applications*, pages 137-149, North-Holland, Amsterdam, 1979.
- [19] Thomann & J.A. Mueller, *Principle of Surface Water Quality Modeling and Control*, Haper &Row, New York, 1987.
- [20] P. Werbos, *Beyond regression: New Tools for Prediction and Analysis in the Behavioural Sciences*, PhD Thesis, Harvard University, 1974.
- [21] D.A. Wismer et al., *Introduction to Nonlinear Optimization: a Problem Solving Approach*, chapter 6, pages 139-162, North-Holland Publishing Company, 1978.