Hybrid Neuro-Fuzzy Inference Systems and Their Applications for On-line Adaptive Learning of Nonlinear Dynamical Systems[1]

JAESOO KIM AND NIKOLA KASABOV

Abstract—*In this paper, an adaptive neuro-fuzzy system, called HyFIS, is proposed to build and optimise fuzzy models. The proposed model introduces the learning power of neural networks into the fuzzy logic systems and provides linguistic meaning to the connectionist architectures. Heuristic fuzzy logic rules and input-output fuzzy membership functions can be optimally tuned from training examples by a hybrid learning scheme composed of two phases: the phase of rule generation from data, and the phase of rule tuning by using the error backpropagation learning scheme for a neural fuzzy system. In order to illustrate the performance and applicability of the proposed neuro-fuzzy hybrid model, extensive simulation studies of nonlinear complex dynamics are carried out. The proposed method can be applied to on-line incremental adaptive learning for the purpose of prediction and control of nonlinear dynamical systems.*

Keywords—Neuro-fuzzy systems, Neural networks, Fuzzy logic, Parameter and structure learning, Knowledge acquisition, Adaptation, Time series.

---

[1]TR-99-05, Department of Information Science, University of Otago, PO Box 56, Dunedin, New Zealand.

Submitted and revised to Neural Networks, 15 March, 1999.

1 INTRODUCTION

Over the last decade or so, significant advances have been made in two distinct technological areas: fuzzy logic and neural networks. The theory of fuzzy logic (Zadeh, 1965) provides a mathematical framework to capture the uncertainties associated with human cognitive processes, such as thinking and reasoning. It provides a mathematical morphology to emulate certain perceptual and linguistic attributes associated with human cognition.

The neural networks paradigm has evolved in the process of understanding the learning and adaptive features of neuronal mechanisms inherent in certain biological species. Neural networks replicate, on a small scale, some of the computational operations observed in biological learning and adaptation. The input-output mapping capability of multilayer perceptron networks allows for a versatile data-driven interpolation derived from numerical examples. One of the drawbacks of such systems is the lack of adequate rule or knowledge extraction (knowledge acquisition), or explanation facilities. Human experts can typically formulate rules describing underlying causual relationships involved in the decision-making process. Traditional neural network-based decision tools, however, do not inherently possess such features. To circumvent these obstacles, integrated techniques between rule-based systems, fuzzy logic and neural network models are being employed for extracting linguistic information from trained network models, or from numerical examples.

The synergism of integrating Fuzzy Logic (FL) systems and Neural Networks (NNs) into a functional system with learning capability and high-level thinking and reasoning will provide a more suitable tool for solving the behaviour of imprecisely-defined complex systems. Considerable work has been done to integrate the greater learning capability of neural networks with fuzzy inference systems (FIS) for deriving the initial rules of a fuzzy system and tune the mem-

bership functions (Lin & Lee, 1991; Berenji & Khedkar, 1992; Horikawa *et al.*, 1992; Yamakawa *et al.*, 1992; Jang, 1993; Hung, 1993; Ishibuchi *et al.*, 1994; Shann & Fu, 1995; Kasabov, 1996; Kasabov *et al.*, 1997; Pal, 1998). These neuro-fuzzy systems have the potential to capture the benefits of two powerful paradigms, fuzzy logic and neural networks, into a single capsule and they have several features that make them well suited to a wide range of knowledge engineering and scientific applications (see for example, Constantin, 1995). These strengths include fast and accurate learning, good generalisation capabilities, excellent explanation facilities in the form of semantically meaningful fuzzy rules, and the ability to accommodate both data and existing expert knowledge about the problem under consideration.

In this study, we propose a general method for combining both numerical and linguistic information into a common framework called HyFIS (Hybrid neural Fuzzy Inference System), in which a two-phase learning scheme is developed as illustrated in Fig. 1. In the first phase, realised in the *knowledge acquisition module*, a method for deriving fuzzy rules from desired input-output data pairs, proposed by Wang & Medal (1992), is used to find proper fuzzy logic rules and the initial structure of the neural fuzzy system. In the second phase, a parameter learning technique using a gradient descent learning algorithm is applied to tune membership functions of input-output linguistic variables.

In the HyFIS architecture, the two phases are repeated on any new set of data, thus making the HyFIS model suitable for incremental, on-line learning of dynamical systems. An important feature of HyFIS is that it is adaptable where the membership functions of the fuzzy predicates, as well as the fuzzy rules can adapt and change according to new training data. The adaptation can take place in an incremental, on-line mode.

The next section introduces the basic concepts of fuzzy rule based reasoning

in neural fuzzy inference systems. In Section 3, we discuss a typical modelling problem that includes *structure learning* and *parameter learning.* In Section 4, the principles and the architecture of HyFIS are described. To illustrate the performance and the applicability of the proposed HyFIS model, experimental results on two well-explored benchmark data— a chaotic time series of the Mackey-Glass (Mackey & Glass, 1977), and nonlinear system identification data, the Box and Jenkins gas furnace data (Box & Jenkins, 1970)—are presented in Section 5. Conclusions and directions for future research are summarised in Section 6.

2 FUZZY KNOWLEDGE-BASED STRUCTURES AND FUZZY REASONING

Fuzzy rules are the fundamental part of a knowledge base that resides in a neural fuzzy inference system. Fuzzy if-then rules are expressions of the form *if* x *is* A *then* y *is* B, where x and y are variables and A and B are labels of fuzzy sets characterised by appropriate membership functions. Fuzzy if-then rules are often employed to capture the imprecise mode of reasoning that plays an essential role in the human ability to make decisions in an environment of imprecision and uncertainty. Through the use of linguistic labels and membership functions, a fuzzy if-then rule can easily capture the spirit of the *rule of thumb* used by humans.

The inference operations upon fuzzy if-then rules are known as *fuzzy reasoning.* The general steps of fuzzy reasoning performed in a fuzzy inference system are as follows:

1. Match the input values towards the membership functions in the premise part to obtain the membership values (or compatibility measures) for each linguistic label.

2. Combine through a specific *t*-norm operator (usually multiplication or *min*) the membership values in the premise part to obtain the *firing strength (weight)* of each rule.

3. Generate the qualified consequent value (either fuzzy or crisp) of each rule depending on the firing strength.

4. Aggregate the qualified consequent values from all rules to produce a crisp output. (This step is called *defuzzification*). Since for the purpose of dynamic system modelling the output of the decision engine should be a crisp value, several methods for defuzzification have been proposed (Lee, 1990). Among these methods, the *centroid* method has been shown to be the most effective (Zadeh, 1965; Kosko, 1992).

Fuzzy inference systems are also known as *fuzzy-rule-based systems, fuzzy models, fuzzy associative memories (FAM), or fuzzy controllers* when used to control dynamical processes (Lee, 1990; Jang, 1993). The main functional blocks of a neural fuzzy inference system are depicted in Fig. 2.

Several types of fuzzy reasoning in a connectionist structure have been proposed in the literature (Lee, 1990; Kosko, 1992; Lin & Lee, 1996; Kasabov, 1996; Jang, Sun, & Mizutani, 1997). Depending on the types of fuzzy reasoning and fuzzy if-then rules employed, most neuro-fuzzy inference systems can be classified into three types (Jang, 1993): Mamdani type, Tsukamoto type, and Takagi-Sugeno type. Fuzzy rules with *certainty factors* (CFs) (Shann & Fu, 1995; Kasabov, 1996a; Pal, 1998) as well as *generalised fuzzy production rules* have also been utilised (Kasabov, 1996a) to capture expert knowledge. These five types of fuzzy reasoning rules are described below.

Type 1 (Mamdani fuzzy models): The overall fuzzy output is derived by applying the *MAX* operation to the qualified fuzzy outputs (each of which is

equal to the minimum of firing strength applied to the output membership functions of each rule). Various schemes have been proposed to choose the final crisp output based on the overall fuzzy output (Mamdani, 1975). An example of a multi-input-single-output (MISO) *Mamdani* fuzzy model can be expressed as:

R_i : if x is A_i, ..., and y is B_i, then $z = C_i$,

where $x, \ldots, y$, and z are linguistic variables representing the process state variables and the control variable, respectively, and $A_i, \ldots, B_i$, and C_i $(i = 1, 2, \ldots, n)$ are the linguistic values of the linguistic variables $x, \ldots, y$, and z in the universe of discourse $U, \ldots, V$, and W, respectively.

Type 2 (Tsukamoto fuzzy models): This is a simplified method based on fuzzy reasoning of Type 1 in which the consequent is required to be *monotonic*; that is, the output membership functions used in this scheme must be monotonic functions (Tsukamoto, 1979). The overall output is the weighted average of each rule's crisp output induced by the rule's firing strength (the product or minimum of the degrees of match with the premise part) and output membership functions.

Type 3 (Takagi-Sugeno fuzzy models): This type of fuzzy if-then rules was proposed by Takagi & Sugeno (1983). The consequent of a fuzzy rule is a function of input linguistic variables. The output of each rule is a linear combination of input variables plus a constant term, and the final output is the weighted average of each rule's output. The following is an example of a multi-input-single-output (MISO) *Takagi-Sugeno* fuzzy model:

R_i : if x is $A_i, \ldots$, and y is B_i, then $z = f_i(x, \ldots, y)$,

where $f_i(x, \ldots, y)(i = 1, 2, \ldots, n)$ is a function of the input variables $x, \ldots, y$.

Type 4 (Fuzzy rules with CFs): This type of a fuzzy rule contains a certainty degree as a *certainty factor* (CF) (or also called weight), e.g.:

R_i : if x is A, then y is B (CF_i),

where CF_i is the certainty factor of the ith rule. For different interpretations and applications of this type of fuzzy rules, see Shann & Fu (1995), Lin & Lee (1996) and Pal (1998).

Type 5 (Generalised production rules): In this type of fuzzy rules several parameters that capture uncertainty are used: (1) *confidence factors* or *certainty factors* (CFs) attached to the conclusion part; (2) relative *degrees of importance* (DI) of the condition elements in the antecedent part; (3) *Noise tolerance* and (4) *sensitivity factor* coefficients (Kasabov, 1996a). Two exemplar fuzzy rules that contain relative *degrees of importance* (DI) of the condition elements and *certainty factors* (CFs) of the conclusion parts are shown below.

R_1 : if x_1 is A_1 ($DI_{1,1}$) and x_2 is B_1 ($DI_{2,1}$) then y is C_1 (CF_1);

R_2 : if x_1 is A_2 ($DI_{1,2}$) and x_2 is B_2 ($DI_{2,2}$) then y is C_2 (CF_2).

This type of fuzzy rules is used in the FuNN model to define the FuNN structure, to insert rules into this model, and to extract rules from a trained FuNN (Kasabov, 1996a; 1996b; 1996c; Kasabov *et al.*, 1997).

In this study, we have constructed a neuro-fuzzy inference system which is functionally equivalent to the Type-4 of fuzzy inference systems (FIS).

3 LEARNING IN NEURAL FUZZY INFERENCE SYSTEMS

Generally, FIS require two major types of learning: structural learning, and parametric learning. Structural learning is concerned with the structure of the fuzzy inference system, i.e., input and output linguistic variables, the membership functions of the terms, the fuzzy rules. Once a satisfactory structure of the FIS is obtained, the fuzzy model needs to perform parametric tuning, e.g., tuning parameters of the membership functions, such as centres, widths, and slopes, and tuning the weights of the fuzzy logic rules. Hence, the main purpose of a neuro-fuzzy system is to apply neural learning techniques to find and tune the parameters and structure of neuro-fuzzy systems.

There are several ways that structure learning and parameter learning can be combined in a neuro-fuzzy system. Existing techniques for parameter learning and structure learning are described in the next subsections.

3.1 Structure learning for FIS

Identification of fuzzy rules has been one of the most important aspects in the design of FIS. By *structure learning*, we mean the extraction of fuzzy logic rules from numerical training data and the tuning of fuzzy partitions of the input and output spaces. Generally, construction of fuzzy logic rules from numerical data consists of two procedures: 1) fuzzy partitioning of the input space and/or output space, and 2) identification of a fuzzy logic rule for each fuzzy subspace. In the following, we shall discuss some techniques for knowledge acquisitions from numerical examples. Most existing techniques can be divided into either of the following cases:

1. *Fuzzy rule extraction through neural network (NN) techniques.* Unsupervised learning, such as self-organising maps (SOMs) (Kohonen, 1989) and adaptive resonance theory (ART) (Carpenter & Grossberg, 1987, 1988, 1990), which construct internal models that capture regularities in the input vector space, is suitable for structure learning in order to find clusters of data indicating the presence of fuzzy logic rules (See also Kosko, 1992). Supervised learning based on a gradient-descent method (Rumelhart *et al.*, 1986), which requires a teacher to specify the desired output vectors, and competitive learning (Grossberg, 1976; Rumelhart & Zipser, 1985; Kosko, 1990), are also commonly used to determine the fuzzy rules (Kosko, 1992; Kasabov, 1996a).

2. *Fuzzy rule extraction by using fuzzy techniques.* A simple method proposed by Wang & Mendal (1992) for generating fuzzy rules from numerical input-output training data is a one-pass build-up procedure. It avoids the time-consuming training procedures typical for NNs. It has been also shown that this approach to building fuzzy rule systems is capable of approximating any real, continuous function on a compact set.

When SOMs are used for rule extraction, the learned clusters of the input patterns are substantially overlapped. As a result, achieving the necessary decision accuracy becomes difficult. Learning vector quantisation (LVQ) techniques are used to refine the near-optimal decision borders, which can provide fine-tuning of SOMs and enhance the decision process.

A hybrid learning scheme that combines SOM and competitive learning for learning fuzzy rules was proposed by Lin and Lee (1991 & 1996). Some heuristics for rule reduction and combination were also provided. Shann *et al.* (1995) and Pal *et al.* (1998) proposed a learning procedure for acquiring fuzzy rules using error backpropagation (EBP) learning algorithm, which is based on a gradient

descent search in the network: the first phase is a training phase, and the second phase is a rule-pruning phase, i.e., after the error backpropagation training, a rule-pruning process is executed to delete redundant fuzzy rules and obtain a rule base with much smaller size than the initial one.

3.2 Parameter learning for FIS

Here, by *parameter learning* we mean tuning of membership functions and other parameters of fuzzy rules in a FIS. There are several learning methods for parameter learning: gradient-descent-based learning algorithms (e.g., the error backpropagation algorithm) (Lin & Lee, 1991; Horikawa *et al.*, 1992; Hung, 1993; Shann & Fu, 1995; Kasabov *et al.*, 1997); reinforcement learning (Berenji & Khedkar, 1992), which requires only a single scalar evaluation of the output; approximate least square estimator (LSE) (Jang, 1993), which is compatible with an extended *Kalman filter* algorithm. The parameter learning method adopted in the HyFIS system is the supervised learning one, with the error backpropagation algorithm.

There are different parameter learning techniques depending on the types of the fuzzy logic rules: 1) fuzzy logic rules with singleton consequents; and 2) Takagi-Sugeno-Kang (TSK) fuzzy rules. Here several neuro-fuzzy techniques for parameter learning of these two types of fuzzy rules are referenced:

1. Parameter learning in multi-input-single-output (MISO) systems is based on the Mamdani fuzzy rules (Type 1). In this case the output membership function is usually bell shaped, triangular, or trapezoidal (Lin & Lee, 1991; Berenji & Khedkar, 1992; Kosko, 1992). The fuzzy response of each rule is defined by its output membership function weighted by its firing strength, and the centroid of the responses is calculated to generate the appropriate output. A defuzzification procedure involves taking the output value with the maximum membership degree as the crisp response of each rule and

then aggregating these responses to produce the appropriate output (Lee, 1990). Almost all existing neuro-fuzzy models for parameter tuning of FIS deal with this type of rules (Horikawa *et al.*, 1992; Hauptmann, & Heesche, 1995; Kasabov 1996a).

2. The other approach in parameter learning is to let the consequent parameter be a linear function of the input variables, instead of a simple constant. In the Takagi-Sugeno-Kang (TSK) fuzzy rule format the response of each rule is a linear combination of the values of the input variables. The responses are then weighted and summed according to the firing strengths to obtain the final output. The consequent of each rule becomes a fuzzy singleton when all the coefficients in the consequent part become zero. That is, when f is a constant (See Type 3 in Section 2), it becomes a zero-order TSK fuzzy model, which can be viewed either as a special case of the Mamdani fuzzy model (Type 1), in which each rule's consequent is specified by a fuzzy singleton, or a special case of the Tsukamoto fuzzy model (Type 2), in which each rule's consequent is specified by a membership function of a step function centre at the constant. This type of fuzzy rules is dealt with in several neuro-fuzzy systems for FIS parameter learning, but it combines optimising the premise membership functions by gradient descent with optimising the consequent equations by linear least squares estimation (Jang, 1993 & 1997).

In the HyFIS model presented in the next section, the fuzzy logic approach to generating fuzzy rules from input-output data pairs is used and implemented in the *knowledge acquisition module* in Fig. 1. After a set of fuzzy rules is extracted, the neural network structure is established, and the second learning phase is performed to adjust the parameters of the membership functions optimally.

4 HyFIS: HYBRID NEURAL FUZZY INFERENCE SYSTEM

The learning procedure in HyFIS consists of two phases. The first phase is the structure learning (rule finding) phase using the *knowledge acquisition module.* The second phase is the parameter learning phase for tuning membership functions to achieve a desired level of performance. One advantage of this approach is that it is very easy to modify the fuzzy rule base as new data become available (Wang & Mendal, 1992). When a new data pair becomes available, a rule is created for this data pair and added to the fuzzy rule base (See Fig. 1). In the second learning phase, the neuro-fuzzy model in the HyFIS uses a multi-layered perceptron (MLP) network based on a gradient descent learning algorithm. The architecture facilitates learning from data and approximate reasoning, as well as knowledge acquisition. It allows for the combination of both numerical data and fuzzy rules, thus producing the synergistic benefits associated with the two sources. In addition, it allows for adaptive learning in a dynamically changing environment. A brief introduction of the structure of the proposed HyFIS system and the functions of each layer is presented in the following subsections.

4.1 The Architecture of HyFIS

The proposed neuro-fuzzy model in the HyFIS is a multilayer neural network-based fuzzy system. Its topology is shown in Fig. 3 and the system has a total of five layers. In this connectionist structure, the input and output nodes represent the input states and output control/decision signals respectively, and in the hidden layers, there are nodes functioning as membership functions (MFs) and rules. This eliminates the disadvantage of a normal feedforward multi-layer net which is difficult for an observer to understand or to modify.

Nodes in layer 1 are input nodes that directly transmit input signals to the next layer. Nodes in layer 2 and 4 are *term nodes* that act as membership functions (MF) to express the input/output fuzzy linguistic variables. Bell-shaped MF functions are used here, in which the mean value c and the variance σ is adapted through the learning process. The fuzzy sets defined for the input/output variables in Fig. 3 are large (L), medium (M), and small (S). But sometimes more granularity is required in some applications such as large positive (LP), small positive (SP), zero (ZE), small negative (SN), and large negative (LN). Each node of layer 3 is a *rule node* and represents one fuzzy rule. The connection weights between layers 3 and 4 represent *certainty factors* (CFs) of the associated rules, i.e., each rule is activated to a certain degree controlled by the weight values.

In the following, special emphasis is placed on how to adapt the parameters that represent the bell-shaped MF for each node in layer 2 and 4 through learning and a detailed description of the components of the HyFIS model's structure and functionalities, and the philosophy behind this architecture are given. Throughout the simulation examples presented in this paper, all the MF used are bell-shaped (Gaussian) functions defined in Eq. (1):

$$\mu_A(x) = Gaussian(x; c, \sigma) = e^{-\frac{(x-c)^2}{\sigma^2}}, \tag{1}$$

A Gaussian membership function is determined by c and σ: c represents the centre of the MF; and σ determines the width of the MF.

The semantic meaning and functions of the nodes in the proposed network are as follows. We use indices i, j, k, and l for nodes in layers 2, 3, 4, and 5, respectively. The output from the nth node of layer m will be denoted by y_n^m.

Layer 1: Nodes in layer one are input nodes that represent input linguistic variables as crisp values. The nodes in this layer only transmit input values to the next layer, the membership function layer. Each node is connected to

only those nodes of layer 2 which represent the linguistic values of corresponding linguistic variable.

Layer 2: Nodes in this layer act as membership functions to represent the terms of the respective linguistic variables. The input values are fed to the layer 2 which calculates the membership degrees. This is implemented using *Gaussian* membership functions with two parameters, centre (or mean, c) and width (or variance, σ). Initially a connection weight in this layer is the unity and the membership functions are spaced equally over the weight space, although if any expert knowledge is available this can be used for initialisation.

The output function of this node is the degree to which the input belongs to the given membership function:

$$y_i^2 = e^{-\frac{(x-c)^2}{\sigma^2}}, \tag{2}$$

where σ and c are the parameters. As the values of these parameters change, the bell-shaped functions vary, thus exhibiting various forms of membership functions on linguistic label. Parameters in this layer are referred to as *precondition parameters.* The input weight represents the centre for that particular MF, with the minimum and maximum determined as the adjacent membership centres.

Layer 3: Each node in layer 3 represents a possible IF-part of a fuzzy rule. The weights of the links are set to unity. The nodes in this layer perform the *AND* operation. Thus, all the nodes in this layer form a fuzzy rule base.

Hence the functions of the layer are as follows.

$$y_j^3 = \min_{i \in I_j}(y_i^2), \tag{3}$$

where I_j is the set of indices of the nodes in layer 2 that are connected to node j in layer 3, and y_i^2 is the output of node i in layer 2.

Layer 4: A node in layer 4 represents a possible THEN-part of a fuzzy rule and each node of this layer performs the fuzzy *OR* operation to integrate the field rules leading to the same output linguistic variables. The nodes of layer 3 and layer 4 are fully connected. In this layer links define the consequences of the rules and a node represents a fuzzy label from the fuzzy quantisation space of an output variable. The activation of the node represents the degree to which this membership function is supported by all fuzzy rules together. The connection weights w_{kj} of the links connecting nodes k in layer 4 to nodes j in layer 3 represent conceptually the *certainty factors* (CFs) of the corresponding fuzzy rules when inferring fuzzy output values. The initial connection weights of the links between layer 3 and 4 are randomly selected in the interval [-1,+1].

The functions of this layer are expressed as follows:

$$y_k^4 = \max_{j \in I_k}(y_j^3 w_{kj}^2), \tag{4}$$

where I_k is the set of indices of the nodes in layer 3 that are connected to the node k in layer 4. Actually these connecting links function as a connectionist inference engine, which avoids the the rule-matching process. Each of the rules is activated to a certain degree represented by the squared weight values.

Layer 5: It represents the output variables of the system. These nodes and links attached to them act as a defuzzifier. A node in this layer computes a crisp output signal. Here the *Centre of Gravity* (COG) or *Centre of Area* (COA) method was used. The centre of area defuzzification scheme, in which the fuzzy centroid constitutes the output signal, can be simulated by

$$y_l^5 = \frac{\sum_{k \in I_l} y_k^4 \sigma_{lk} c_{lk}}{\sum_{k \in I_k} y_k^4 \sigma_k}, \tag{5}$$

where I_l is the set of indices of the nodes in layer 4 which are connected to the node l in layer 5 and c_{lk} and σ_{lk} are respectively the centroid and width of the membership function of the output linguistic value represented by the k in layer 4. The weights of the links from the nodes in layer 5 to the nodes in layer 4 are unity. Thus the only learnable weights in the network are w_{kj}s between layer 3 and layer 4.

4.2 Hybrid Learning Algorithms for HyFIS

In this section we present the two-phase hybrid learning scheme for the proposed HyFIS model. In phase one, *rule finding phase*, fuzzy techniques are used to find the rules. In phase two, a supervised learning scheme based on gradient descent learning is used to optimally adjust the MF for desired outputs. To initiate the learning scheme, training data and the desired or guessed coarse of fuzzy partition, i.e., the initial size of the term set of each input/output linguistic variable, must be provided from the outside world.

4.2.1 A general learning scheme for HyFIS

The following procedure outlines the adaptive, incremental training of the HyFIS model:

Step 0: Initialise the neuro-fuzzy model in the HyFIS. To initiate the learning scheme, training data and the desired or guessed coarse of fuzzy partition, i.e., the size of the term set of each input-output linguistic variables, are given. The initial weights of the links between layer 3 and 4 are randomly selected in [-1,+1].

Step 1: Extract a set of fuzzy rules from input-output data set using the first-phase learning method as described in the next section.

Step 2: Update the rules and if necessary, add new fuzzy rules into the neuro-fuzzy structure: for each fuzzy rule, check if the rule is already represented in the network structure, if yes, then update it; if not, then add this rule to the network structure.

Step 3: Apply parameter learning on the data set as well as on old data if available or if necessary. This step may not be performed on each individual new data item, but on a set of data depending on the frequency of the incoming data stream.

Step 4: Repeat from Step 1.

4.2.2 Rule Finding Phase

We consider a simple and straightforward method proposed by Wang & Mendal (1992) for generating fuzzy rules from numerical input/output training data.

The task here is to generate a set of fuzzy rules from the desired input/output pairs and then use these fuzzy rules to determine the structure of the Neuro-Fuzzy system in the HyFIS environment. To illustrate this method, suppose we are given a set of desired input/output data pairs: $(x_1^1, x_2^1; y^1), (x_1^2, x_2^2; y^2), \ldots$, where x_1 and x_2 are inputs and y is the output. It is straightforward to extend this method to general multi-input-multi-output (MIMO) cases. A set of desired input-output data pairs, the simple two-input one-output case is chosen in order to emphasise and to clarify the basic ideas of the methodology. This approach consists of the following three steps:

Step 1: Divide the input and output space into fuzzy regions. After the number of membership functions associated with each input and output are fixed, the initial values of parameters are set in such a way that the centres of the membership functions are equally spaced along the range of each input

and output variable. Moreover, these membership functions satisfy the condition of ϵ-*completeness* (Lee, 1990) with $\epsilon = 0.5$, 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 $\mu_A(x) \geq \epsilon$. In this manner, the fuzzy inference system can provide smooth transitions and sufficient overlap from one linguistic label to another. So, one has to choose the intervals for the linguistic values of each input and output linguistic variable in such a way that they do overlap and also cover the entire space of the corresponding input-output linguistic variables.

For instance, assume that the domain intervals of x_1, x_2, and y are [-1,1], Divide each domain interval into N regions and assign each region a fuzzy membership function. Figure 4 shows an example where the domain interval of x_1, x_2, and y are divided into five regions. Of course, other divisions of the domain regions and other shapes of membership functions are possible.

Step 2: Generate fuzzy rules from given data pairs. First, determine the degrees of given data pairs in different regions. For example, suppose we are given the following set of desired input-output data pairs:

$$(0.6, -0.2; 0.2), (0.4, 0; 0.4), \cdots \tag{6}$$

where the first two numbers (x_1, x_2) are inputs and third one (y) is the output for each data pair. For example, x_1^1 in Fig. 4 has degree 0.8 in SP, degree 0.2 in LP, Similarly, x_2^2 has degree 1 in ZE and smaller degrees than x_2^2 in all other regions. Second, assign x_1^i, x_2^i, and y^i to a region with maximum degree: for example, x_1^1 in Fig. 4 is assigned to SP and x_2^2 in Fig. 4 is assigned to ZE. Finally, obtain one rule from one pair of desired input/output data, for example,

$(x_1^1, x_2^1; y^1) \Longrightarrow [x_1^1(0.8\ in\ SP), x_2^1(0.6\ in\ ZE), y^1(0.6\ in\ ZE)]$,

- R_1: if x_1 is SP and x_2 is ZE, then y is ZE;

$(x_1^2, x_2^2; y^2) \Longrightarrow [x_1^2(0.8\ in\ SP), x_2^2(1\ in\ ZE), y^2(0.8\ in\ SP)]$,

- R_2: if x_1 is SP and x_2 is ZE, then y is SP.

Step 3: Assign a degree to each rule. To resolve a possible conflict problem, i.e., rules having the same IF-part but a different THEN-part, and to reduce the number of rules, we assign a degree to each rule generated from data pairs and accept only the rule from a conflict group that has a maximum degree. In other words, this step is performed to delete redundant rules for obtaining a concise fuzzy rule base and the effects of these rules are that each of the rules is activated to a *certain degree* represented by the weight value (the certainty factor) associated with that rule.

The following product strategy is used to assign a degree to each rule: The degree of the rule denoted by

R_i : if x_1 is A and x_2 is B, then y is C (w_i),

is defined as

$$w_i = \mu_a(x_1)\mu_b(x_2)\mu_c(y). \tag{7}$$

For example, R_1 has a degree of

$$\begin{aligned} w_1 &= \mu_{SP}(x_1)\mu_{ZE}(x_2)\mu_{ZE}(y) \\ &= 0.8 \times 0.6 \times 0.6 \\ &= 0.288, \end{aligned}$$

and R_2 has a degree of

$$\begin{aligned} w_2 &= \mu_{SP}(x_1)\mu_{ZE}(x_2)\mu_{SP}(y) \\ &= 0.8 \times 1 \times 0.8 \\ &= 0.64. \end{aligned}$$

Note that if two or more generated fuzzy rules have the same preconditions and consequents, then the rule that has maximum degree is used. In this

way, assigning the degree to each rule, the fuzzy rule base can be adapted or updated by the relative weighting strategy: the more task-related the rule becomes, the more weight degree the rule gains. As a result not only is the conflict problem resolved, but also the number of rules is greatly reduced.

4.2.3 Parameter Learning Phase

After the fuzzy rules are found, the whole network structure is established, and the network then enters the second learning phase to adjust optimally the parameters of the membership functions.

Let d_l be the target output of the node l in layer 5 for an input vector $X = (x_1, x_2, ..., x_p)$. We now derive the learning algorithm for the HyFIS with node functions defined in the previous section using a gradient descent learning algorithm to minimise the error function,

$$E = \frac{1}{2} \sum_{X} \sum_{l=1}^{q} (d_l - y_l^5)^2, \tag{8}$$

where q is the number of nodes in layer 5 and d_l and y_l are the target and actual outputs of the node l in layer 5 for an input X.

Assuming that the weight w_{kj} is adjustable parameter in the node k in layer 4 to the node j in layer 3 (e.g., c and σ in this learning phase), the general learning rule used by a gradient descent learning is

$$w_{kj}\ (t+1) = w_{kj}(t) - \eta\ (\frac{\partial E}{\partial w_{kj}}), \tag{9}$$

where $\eta > 0$ is the learning rate, and the *chain rule* is described as follows:

$$\begin{aligned} \frac{\partial E}{\partial w_{kj}} &= \frac{\partial E}{\partial y_k^4} \frac{\partial y_k^4}{\partial w_{kj}} \\ &= \frac{\partial E}{\partial y_l^5} \frac{\partial y_l^5}{\partial y_k^4} \frac{\partial y_k^4}{\partial w_{kj}}. \end{aligned} \tag{10}$$

To illustrate the learning rule for each parameter, we shall describe the computations of $\frac{\partial E}{\partial w_{kj}}$, layer by layer, starting at the output nodes, and we will use

Gaussian membership functions with centres (c) and widths (σ) as adjustable parameters for these computations.

The learning rules of each layer are derived below:

Layer 5: From Eq.(8) we get,

$$\frac{\partial E}{\partial y_l^5} = -(d_l - y_l^5). \tag{11}$$

Hence, the error to be propagated to the preceding layer is

$$\delta_l^5 = -\frac{\partial E}{\partial y_l^5} = d_l - y_l^5. \tag{12}$$

Using Eq. (10), the adaptive rule of the variance (width, σ) is derived as:

$$\frac{\partial E}{\partial \sigma_{lk}} = \frac{\partial E}{\partial y_l^5}\frac{\partial y_l^5}{\partial \sigma_{lk}}. \tag{13}$$

Recalling Eq.(5), $y_l^5 = \frac{\sum(y_k^4\ \sigma_{lk}\ c_{lk})}{\sum(y_k^4\ \sigma_{lk})}$, we obtain:

$$\begin{aligned}\frac{\partial y_l^5}{\partial \sigma_{lk}} &= \frac{\left(\sum_{k'} y_{k'}^4 \sigma_{lk'}\right)\ y_{lk}\ c_{lk} - \left(\sum_{k'} y_{k'}^4 \sigma_{lk'} c_{lk'}\right)\ y_{lk}}{\left(\sum_{k'} y_{k'}^4 \sigma_{lk'}\right)^2} \\ &= \frac{y_{lk}\left(c_{lk}\left(\sum_{k'} y_{k'}^4 \sigma_{lk'}\right) - \left(\sum_{k'} y_{k'}^4 \sigma_{lk'} c_{lk'}\right)\right)}{\left(\sum_{k'} y_{k'}^4 \sigma_{lk'}\right)^2}.\end{aligned} \tag{14}$$

Here k' is the index of a node in layer 4 which is connected to a node l in layer 5.

Using Eq.(11), and (14), we can write Eq.(13) as

$$\begin{aligned}\frac{\partial E}{\partial \sigma_{lk}} &= \frac{\partial E}{\partial y_l^5}\frac{\partial y_l^5}{\partial \sigma_{lk}} \\ &= -(d_l - y_l^5)\frac{y_{lk}\left(c_{lk}\left(\sum_{k'} y_{k'}^4 \sigma_{lk'}\right) - \left(\sum_{k'} y_{k'}^4 \sigma_{lk'} c_{lk'}\right)\right)}{\left(\sum_{k'} y_{k'}^4 \sigma_{lk'}\right)^2}.\end{aligned} \tag{15}$$

Hence, the σ parameter is updated by

$$\sigma_{lk}(t+1) = \sigma_{lk}(t) + \eta\delta_l^5 \frac{y_{lk}\left(c_{lk}\left(\sum_{k'} y_{k'}^4\sigma_{lk'}\right) - \left(\sum_{k'} y_{k'}^4\sigma_{lk'}c_{lk'}\right)\right)}{\left(\sum_{k'} y_{k'}^4\sigma_{lk'}\right)^2}. \quad (16)$$

Similarly, the adaptive rule of the mean (centre, c) is derived as

$$\begin{aligned}\frac{\partial E}{\partial c_{lk}} &= \frac{\partial E}{\partial y_l^5}\frac{\partial y_l^5}{\partial c_{lk}} \\ &= -(d_l - y_l^5)\frac{\sigma_{lk}y_k^4}{\sum_k \sigma_{lk}y_k^4}. \end{aligned} \quad (17)$$

Hence, the c parameter is updated by

$$c_{lk}(t+1) = c_{lk}(t) + \eta\delta_l^5\frac{\sigma_k y_k^4}{\sum_k \sigma_{lk}y_k^4}. \quad (18)$$

Layer 4: The error for nodes in this layer is calculated based on fuzzification of desired outputs and activation of each node. Only the error signals need to be computed and propagated. Hence we have

$$\delta_k^4 = \frac{\partial E}{\partial y_k^4} = \frac{\partial E}{\partial y_l^5}\frac{\partial y_l^5}{\partial y_k^4}. \quad (19)$$

From Eq.(8) we get,

$$\frac{\partial E}{\partial y_l^5} = -(d_l - y_l^5) \quad (20)$$

From Eq. (5) we get,

$$\frac{\partial y_l^5}{\partial y_k^4} = \frac{\sigma_{lk}\left(c_{lk}\left(\sum_{k'} y_{k'}^4\sigma_{lk'}\right) - \left(\sum_{k'} y_{k'}^4\sigma_{lk'}c_{lk'}\right)\right)}{\left(\sum_{k'} y_{k'}^4\sigma_{lk'}\right)^2}. \quad (21)$$

Hence, using Eqs. (11) and (21), we can write the error signal of Eq. (19) as

$$\delta_k^4 = (d_l - y_l^5)\frac{\sigma_{lk}\left(c_{lk}\left(\sum_{k'} y_{k'}^4\sigma_{lk'}\right) - \left(\sum_{k'} y_{k'}^4\sigma_{lk'}c_{lk'}\right)\right)}{\left(\sum_{k'} y_{k'}^4\sigma_{lk'}\right)^2}. \quad (22)$$

Layer 3: As in layer 4, no parameters need to be adjusted in this layer, and only the error signal needs to be computed and propagated backwards. The error signal δ_j^3 is derived as following:

$$\begin{aligned} \delta_j^3 &= \frac{\partial E}{\partial y_j^3} = \frac{\partial E}{\partial y_k^4}\frac{\partial y_k^4}{\partial y_j^3} = \frac{\partial E}{\partial y_k^4}, \\ &= \delta_k^4. \end{aligned} \tag{23}$$

Layer 2: If input values lie in the fuzzy segment, then the corresponding parameters should be increased directly proportional to the propagated error from the previous layer, because the error is caused by these parameters.

Using (10) and (2), the adaptive rule of c_i is derived as in the following:

$$\begin{aligned} \frac{\partial E}{\partial c_i} &= \frac{\partial E}{\partial y_i^2}\frac{\partial y_i^2}{\partial c_i} \\ &= \frac{\partial E}{\partial y_i^2} y_i^2 \frac{2(x-c_i)}{\sigma_i^2}, \end{aligned} \tag{24}$$

where from (3)

$$\frac{\partial E}{\partial y_i^2} = \frac{\partial E}{\partial y_j^3}\frac{\partial y_j^3}{\partial y_i^2}, \tag{25}$$

where from Eq. (23)

$$\frac{\partial E}{\partial y_j^3} = \delta_j^3 \tag{26}$$

and from Eq.(3) and Eq. (23)

$$r = Arg \min_{i \in I_j}(y_i^2), \tag{27}$$

Then,

$$\begin{aligned} \frac{\partial E}{\partial y_i^2} &= \sum_j \delta_j^3 \qquad if \quad i = r \\ &= 0 \qquad otherwise. \end{aligned} \tag{28}$$

So the mean (c_i^2) of the input membership functions can be updated by

$$c_i(t+1) = c_i(t) + \eta \frac{\partial E}{\partial y_i^2} y_i^2 \frac{2(x-c_i)}{\sigma_i^2}, \tag{29}$$

Similarly, using (10) and (2), the adaptive rule of σ_i^2 is derived as

$$\begin{aligned} \frac{\partial E}{\partial \sigma_i} &= \frac{\partial E}{\partial y_i^2}\frac{\partial y_i^2}{\partial \sigma_i} \\ &= \frac{\partial E}{\partial y_i^2} y_i^2 \frac{2(x-c_i)^2}{\sigma_i^3}. \end{aligned} \tag{30}$$

Hence, the variance (σ_i) of the input membership functions now become

$$\sigma_i(t+1) = \sigma_i(t) + \eta \frac{\partial E}{\partial y_i^2} y_i^2 \frac{2(x-c_i)^2}{\sigma_i^3}. \tag{31}$$

5 APPLICATION TO NONLINEAR DYNAMICAL SYSTEMS

Non-linear dynamical time-series modelling is a generic problem which permeates all fields of science. Applications of time-series prediction can be found in the areas of economic and business planning, inventory and production control, weather forecasting, signal processing, control, and lots of other fields. The increased interest in nonlinear systems is also related to the discovery of *chaos*, as chaos can readily occur in all natural and living systems where *nonlinearity* is present. Hence, chaos is currently one of the most exciting topics in nonlinear systems research. This section describes two applications of the proposed methodology.

5.1 Example 1–Nonlinear System Identification of the Box-Jenkins Time Series

System identification uses a techniques that permits to build mathematical models of dynamical systems based on input-output data. In this subsection we apply the HyFIS to nonlinear system identification, using the well-known Box and Jenkins gas furnace data (Box & Jenkins, 1970). This data set was recorded from a combustion process of a methane-air mixture. During the process the portion

of methane was randomly changed, keeping a constant gas flow rate. This is a time-series data set for a gas furnace process with gas flow (the inlet methane) rate, $u(t)$, as the furnace input, and CO_2 concentration in outlet gas, $y(t)$, as the furnace output. We assume that the task of the model is

$$(u(t-\tau_1), y(t-\tau_2)) \Rightarrow y(t), \tag{32}$$

where delays τ_1 and τ_2 should be determined beforehand.

5.1.1 Experimental Design

The task of the HyFIS model is to provide an identification for the CO_2 concentration $y(t)$ given the methane gas portion from four time steps before $u(t-4)$ and the last CO_2 concentration $y(t-1)$. Fig. 5 shows a schematic diagram for the process of identification using the HyFIS for the case where the fuzzy sets defined for the input-output variables are S (small); M (medium); and L (large) in this case.

The original data set consists of 296 $[u(t), y(t)]$ data pairs that were converted so that each training data point consists of $[u(t-4), y(t-1); y(t)]$ which reduces the number of data points to 292. For this specific problem, we use the MF shown in (a) in Fig. 8. Each domain interval was divided into five linguistic labels, and the fuzzy regions denoted by VS (very small), S (small), M (medium), L (large), and VL (very large). We used the three-step procedure of Section 4 to generate the fuzzy rule base, based on 292 examples. The fuzzy rules generated from the desired input-output pairs and their corresponding degrees are given in Table 1. After the fuzzy logic rules have been found, the network structure is established as in Fig. 6.

In this example we consider the situation where neither linguistic fuzzy rules alone nor desired input-output pairs alone are sufficient for a successful prognosis to a desired accuracy. A combination of linguistic fuzzy rules and fuzzy

rules generated from the desired input-output data pairs is, however, sufficient to successfully predict the level of CO_2 to the desired level of accuracy.

5.1.2 Experimental Results

We consider two cases where the first part of the information comes from desired input-output pairs, whereas the second part of the information comes from linguistic rules. To do this we firstly selected 200 data pairs randomly and used them to generate the fuzzy rule base, which is the same as Table 1 except that there are no rules in the Table outlined by the '$*$'. With these fuzzy logic rules the network structure is established as in Fig. 6 except that there are no connections illustrated by the empty circles ($\circ$). The fuzzy rule base of linguistic rules for the remaining 92 data pairs was chosen to have only two rules marked by '$*$' in Table 1. In this case, the whole network structure is showed only with the connections illustrated by the *empty* circles ($\circ$) in Fig. 6.

We simulated the following three cases: 1) 200 training data examples were used to construct the fuzzy rule base and the HyFIS model is structured by this fuzzy rule base, and the parameter learning phase is applied; 2) the fuzzy rule base of selected linguistic rules from 92 data pairs and the structure of the HyFIS is established with only this fuzzy rule base, and the parameter learning phase is applied; and 3) the fuzzy rule base which combined the fuzzy rule bases of 1 and 2 cases is used to establish the whole network structure, and the initialised network is trained to tune the parameters of the model optimally to achieve a desired level of performance. For each of the cases, the data were partitioned in 200 data points as the training set, the remaining 92 points as the test set for validation.

The experiment results for the first two cases are shown in Fig. 7. After 200 epochs of training, root mean square errors of $RMSE_train = 0.0382$ and

$RMSE_test = 0.0588$ were obtained. The initial and final membership functions are shown in Fig. 8 and 9, respectively. Finally in Fig. 10 and 11, the third case was described. We see very clearly from these figures that, for the first and second cases the system cannot be predicted the level of CO_2 to the desired level of accuracy, whereas for case 3 we successfully identified the desired nonlinear systems dynamics with an RMSE = 0.0205.

5.2 Example 2–Prediction of the Mackey-Glass Chaotic Dynamics

The chaotic time series used in our simulation is generated by a delay differential equation

$$\frac{dx(t)}{dt} = \frac{\alpha x(t-\tau)}{1+x^{\gamma}(t-\tau)} - \beta x(t) \tag{33}$$

that was first investigated by Mackey and Glass (Mackey & Glass, 1977). Keeping the parameters α, β and γ fixed at $\alpha = 0.2, \beta = 0.1$ and $\gamma = 10$ leaves τ as the only adjustable parameter. The behaviour of the Mackey & Glass equation (33) as a function of the delay parameter (τ) has been studied extensively and is reported by Farmer (1982). As τ is varied, the system exhibits either fixed-point, limit cycle, or chaotic behaviour. Choosing $\tau = 17$ yields chaotic behaviour, and a strange attractor (Ott, 1981), with a fractal dimension of approximately 2.1, i.e., $x(t)$ is quasi-periodic and chaotic with a fractal attractor dimension of 2.1. The characteristic time constant of $x(t)$ is 50, which makes difficult to forecast $x(t + \Delta t)$ with $\Delta t > 50$. Meanwhile, $\tau = 30$ yields a strange attractor with the fractal dimension of approximately 3.5. Higher values of τ yield higher dimensional chaos. Note that because of the delay, $x(t-\tau)$, the phase space of this system is infinite dimensional. However, as time progresses the system collapses onto the low dimensional strange attractor. Other infinite dimensional chaotic systems, such as nonlinear partial differential equations, also display collapse

onto low dimensional attractors. Thus, the Mackey-Glass equation (33) exhibits in the simpler setting of nonlinear, differential equations behaviour that occurs in much more complicated systems such as nonlinear partial differential equations. A detailed analysis of the chaotic properties of equation (33) may be found in Ott (1981) & Farmer (1982).

5.2.1 Experimental Design

The goal is to take a set of values of $x(\cdot)$ at discrete times in some time window containing times less than t, and use the values to accurately predict $x(t + \Delta t)$, where Δt is some prediction time step into the future. One may fix Δt, collect statistics on accuracy for many prediction times t by sliding the window along the time series, and then increase Δt and again collect statistics on accuracy. It can be observed how an average index of accuracy changes as Δt is increased. The fundamental nature of chaos dictates that prediction accuracy will decrease as Δt is increased. This is due to inescapable inaccuracies of finite precision in specifying the $x(t)$ at discrete times in the past that are used for predicting the future. Thus, all predictive methods will degrade as Δt is increased. The standard method for this type of prediction is to create a mapping from D points of the time series spaced Δ apart, that is, $(x(t - (D - 1)\Delta), \ldots, x(t - \Delta), x(t))$, to a predicted future value $x(t + \Delta t)$. Embedding a set of time-series values in a state vector is common to several approaches, including those of Lapedes & Farber (1987), and Moody & Darken (1989). The prediction of future values of this time series is a benchmark problem which has been also considered by a number of connectionist researchers (Lapedes & Farber, 1987; Moody & Darken, 1989; Casdagli, 1989; Crowder, 1990; Weigend, 1990).

At $\tau = 17$, $x(t)$ appears to be quasi-periodic and the power spectrum is broadband with numerous spikes due to the quasi-periodicity. At $\tau = 30$, $x(t)$ is

even more irregular. Fig. 12 shows a plot of $x(t)$ *versus time* (t) for a time span of 1000 time steps for $\tau = 17$. If Δ is a time delay, and m is an integer, then our goal is to use the neuro-fuzzy model to construct a function

$$y(t + \Delta t) = f(x(t), x(t - \Delta), x(t - 2\Delta), \ldots, x(t - m\Delta)), \tag{34}$$

where Δt is a prediction time into the future and $f(\cdot)$ is a map. This may be viewed as an $m+1$ dimensional space. Thus, the *embedding dimension* is defined to be $m + 1$. $y(t + \Delta t)$ is the output of a single node in the output layer, and x is inputs that take on $x(t), x(t - \Delta), \ldots, x(t - m\Delta)$. $y(t + \Delta t)$ takes on the value $x(t + \Delta t)$.

We have not yet specified what m and Δ should be. An important theorem of Takens (Takens, 1981) states that if the dimension of the fractal attractor is defined to be, d_f, then an embedding dimension, d_e lies in the range $d_f < d_e < 2d_f + 1$. We therefore choose $d_e = 4$, for $\tau = 17$. Takens provides no information on Δ and the time span we want to forecast into the future is $\Delta t = 6$ for $\tau = 17$.

We extracted 1000 input-output data pairs which consist of four past values of $x(t)$, i.e.,

$$[x(t - 18), x(t - 12), x(t - 6), x(t); x(t + 6)], \tag{35}$$

where $t = 118$ to 1117. There are therefore 4 inputs to the neuro-fuzzy system, representing these values of $x(t)$, and one linear output representing the value $x(t + \Delta t)$. The first 500 (from $x(1)$ to $x(500)$) pairs was used for estimation as the training set, while the remaining 500 pairs (from $x(501)$ to $x(1000)$) were used for validating the identified model as the test set. The number of membership functions (MFs) assigned to each input and output was initially set to 3: S (small); M (medium); and L (large).

5.2.2 Experimental Results for On-line Adaptive Learning

We conducted two experiments. In the first experiment, 500 training data were used to construct a fuzzy rule base and then a HyFIS model is structured accordingly. The HyFIS model was then trained to fine-tune the membership functions to achieve a desired level of performance.

In the second experiment, a fuzzy rule base was generated from the first 500 data points; then the network is trained to tune the parameters and used for a prediction of the next data points $x(501)$; the estimated value of the $x(501)$ data point is then used to update the fuzzy rule base. The updated network structure is then used to predict $x(502)$, etc. This adaptive procedure continued until $x(1000)$.

The updated fuzzy rule base is shown in Table 3. The number of rules are same as in Table 2, but the degrees of each rule and one rule which was illustrated by the '$*$' are changed. The results are shown in Fig. 13. The smallest RMSE value for training is 0.0112, whose curve is depicted in Fig. 13 (a). Comparing (b) and (d) in Fig. 13 we see that the prediction was greatly improved when we used the adaptive fuzzy rule base procedure. One of the advantage of the HyFIS model is that it is easy to modify the fuzzy rule base and the network structure as new data become available. When a new data pair becomes available, we create a rule for this data pair and update the fuzzy rule base and the network structure. The updated network is trained on new data. Then, the neuro-fuzzy system is used to predict future values. Fig. 14 also shows the initial membership functions for each input and output variables, and the final membership functions for input and output variables, respectively, after the network was trained with the fuzzy rule base generated from the first 500 training data.

6 CONCLUDING REMARKS

In this paper, we developed two-phase neuro-fuzzy system called HyFIS to resolve problems of the traditional fuzzy models and neural networks. In the first phase, the knowledge acquisition module using fuzzy techniques can generate proper and concise fuzzy rules from numerical data. This module can be used to combine both numerical and linguistic information into a fuzzy rule base. This fuzzy rule base then provides the initial structure of a neural network. After the structure of the network is established, the network enters the second learning phase to adjust the parameters of the membership functions to achieve a desired level of performance. We applied our new hybrid neuro-fuzzy system to nonlinear dynamical systems, including a chaotic time-series prediction problem.

The main features and advantages of the HyFIS developed in this paper are: 1) It is a general framework that combines two technologies, namely neural networks and fuzzy systems; 2) By using fuzzy techniques, both numerical and linguistic knowledge can be combined into a fuzzy rule base; 3) A combined fuzzy rule base comprises the knowledge of the network structure so that *structure learning* techniques can be easily accomplished; 4) Fuzzy membership functions can be tuned optimally by using a gradient descent learning method.

Advantages of the two-phase neuro-fuzzy hybrid techniques in the HyFIS model also include their non-linearity, the capability of fast learning from numerical and linguistic knowledge, and the adaptation capability. Furthermore, the availability of dedicated neuro-fuzzy processors (Chiaberge, 1995) would permit a very fast processing in HyFIS that would allow its use for real-time applications.

Although the neuro-fuzzy approach provides good performance for nonlinear dynamical systems, it cannot always satisfy all the requirements of real-world applications. In such cases, the chosen modelling strategy must be integrated with other approaches so that the designers of a hybrid intelligent system can

choose among several other paradigms, including evolutionary computing, finite state automata, and conventional AI systems (expert system) according to the system requirements and to what form of knowledge is avaliable about the target system.

The list of potential applications of HyFIS includes: real-time adaptive control; on-line financial data analysis and prediction; real-time adaptive speech and image processing; decision making in a changing environment (e.g., moving objects.) These applications are the subject of our on-going research projects.

Acknowledgements

The authors acknowledge the research grant (UOO808) funded by the Foundation for Research Science and Technology in New Zealand and would like to thank the reviewers for their valuable discussions and suggestions which have helped to improve the quality of this paper.

7 REFERENCES

Berenji, H.R., & Khedkar, P. (1992). Fuzzy Rules for Guiding Reinforcement Learning. *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Mallorca, 511-514.

Box, G.E.P., & Jenkins, G.M. (1970). Time series analysis, forecasting and control. San Francisco: Holden Day.

Carpenter, G.A., & Grossberg, S. (1987). A massively parallel architecture for a self-organising neural pattern recognition machine. *Computer Vision Graphics Image processing*, **37**, 54-115.

Carpenter, G.A., & Grossberg, S. (1988). The ART of adaptive pattern recogni-

tion by a self-organisation neural network. *Computer*, **21**(3), 77-88.

Carpenter, G.A., & Grossberg, S. (1990). ART 3: Hierarchical search using chemical transmitters in self-organising pattern recognition architectures. *Neural Networks*, **3**, 129-152.

Carpenter, G.A., Grossberg, S., & Rosen, D.B. (1991). Fuzzy ART: Fast stable learning and categorisation of analog patterns by an adaptive resonance system. *Neural Networks*, **4**, 759-771.

Carpenter, G.A., Grossberg, S., Reynolds, J.H., & Rosen, D.B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, **3**(5), 698-713.

Casdagli, M. (1989). Nonlinear prediction of chaotic time series. *Physica D*, **35**, 335-356.

Chiaberge, M., Miranda, E., & Reyneri, L.M. (1995). A Pulse Stream for Low-Power Neuro-Fuzzy Computation. *IEEE Transactions on Circuits and Systems*, **42**(11), 946-954.

Constantin, V.A. (1995). Fuzzy Logic and Neurofuzzy Applications Explained. Prentice Hall.

Crowder, R.S. (1990). Predicting the Mackey-Glass time series with cascade-correlation learning. *Proceedings of the 1990 Connectionist Models Summer School*, D. Touretzky, G. Hinton, and T. Sejnowski, Eds., Carnegie Mellon Univ., 117-123.

Farmer, J.D. (1982). Chaotic Attractors of an Infinite-dimensional Dynamical System. *Physica 4D*, **3**, 366-393.

Fu, L.M. (1993). Knowledge-based connectionism for revising domain theories. *IEEE Transactions on System, Man and Cybernetics*, **23**(1), 173-182.

Grossberg, S. (1976). Adaptive pattern classification and universal recording, I: Parallel development and coding of neural feature detectors. *Biological Cybernet-*

ics, **23**, 121-134.

Hauptmann, W., & Heesche K. (1995). A Neural Net Topology for Bidirectional Fuzzy-Neuro Transformation. *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE/IFES)*, Yokohama, Japan, 1511-1518.

Holden, A.D.C., & Suddarth, S.C. (1993). Combined neural-net/knowledge-based adaptive systems for large scale dynamic control. *Neural Networks in Pattern Recognition and Their Applications*, C.H. Chen, Ed., World Scientific, Singapore, 1-20.

Horikawa, S., Furuhashi, T., & Uchikawa, Y. (1992). On fuzzy modelling using fuzzy neural networks with the back-propagation algorithm. *IEEE Transactions on Neural Networks*, **3**(5), 801-806.

Hung, C.C. (1993). Building a Neuro-Fuzzy Learning Control System. *AI Expert*, November, 40-49.

Ishibuchi, H., Tanaka, H., & Okada, H. (1994). Interpolation of Fuzzy If-Then Rules by Neural Networks. *International Journal of Approximate Reasoning*, **10**(1), 3-27.

Jang, J.S.R. (1993). ANFIS: Adaptive–network–based fuzzy inference systems. *IEEE Transactions on System, Man and Cybernetics*, **23**(3), 665-685.

Jang, J.S.R., Sun, C.T., & Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Upper Saddle River, NJ: Prentice-Hall.

Kasabov, N. (1996a). Foundations of Neural Networks, Fuzzy Systems and Knowledge Engineering. Cambridge, MA: MIT Press.

Kasabov, N. (1996b). Learning fuzzy rules and approximate reasoning in fuzzy neural networks and hybrid systems. *Fuzzy Sets and Systems*, **82**, 135-149.

Kasabov, N. (1996c). Adaptable neuro production systems. *Neurocomputing*, **13**, 95-117.

Kasabov, N., Kim, J., Watts, M., & Gray, A. (1997). FuNN/2–A Fuzzy Neural

Network Architecture for Adaptive Learning and Knowledge Acquisition. *Information Sciences*, **101**(3), 155-175.

Kohonen, T. (1989). Self-Organisation and Associative Memory. 3nd Ed., New York: Springer-Verlag.

Kosko, B. (1990). Unsupervised Learning in Noise. *IEEE Transactions on Neural Networks*, **1**(1), 44-57.

Kosko, B. (1992). Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence. Englewood Cliffs, NJ: Prentice-Hall.

Lapedes, A.S., & Farber, R. (1987). Nonlinear signal processing using neural networks: prediction and system modelling. *Tech. Rep. LA-UR-87-2662*, Los Alamos Nat. Lab., Los Alamos, New Mexico.

Lee, C.C. (1990). Fuzzy Logic in Control Systems: Fuzzy Logic Controller—Part I & II. *IEEE Transactions on System, Man and Cybernetics*, **20**(2), 404-435.

Lin, C.T., & Lee, C.S.G. (1991). Neural-Networks-Based Fuzzy Logic Control and Decision System. *IEEE Transactions on Computers*, **40**(12), 1320-1366.

Lin, C.T., & Lee, C.S.G. (1996). Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems. Upper Saddle River, NJ: Prentice-Hall.

Mackey, M., & Glass, L. (1977). Oscillation and chaos in physiological control systems. *Science*, **197**, 287-289.

Mamdani, E.H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, **7**(1), 1-13.

Moody, J. & Darken, C.J. (1989). Fast learning in networks of locally tuned processing units. *Neural Computation*, **1**, 281-294.

Ott, E. (1981). Strange attractors and chaotic motions of dynamical systems. *Review Modern Physics*, **53**(4), 655-671.

Pal, N.R., & Pal, T. (1998). On Rule Pruning using Fuzzy Neural Networks. *Private Communication.*

Rumelhart, D.E., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, **9**, 75-112.

Rumelhart, D.E., & McClelland, J.L. (1986). Parallel Distributed Processing, Cambridge, Mass. : MIT Press.

Shann, J.J., & Fu, H.C. (1995). A fuzzy neural network for rule acquiring on fuzzy control system. *Fuzzy Sets and Systems*, **71**(3), 345-357.

Takagi, T., & Sugeno, M. (1983). Derivation of fuzzy control rules from human operator's control actions. *Proceedings of the IFAC Symposium on Fuzzy Information, Knowledge Representation and Decision Analysis*, 55-60.

Takens, F. (1981). Detecting Strange Attractor in Turbulence. *Lecture Notes in Mathematics*, D. Rand, and L. Young, Eds., Springer Berlin, page 366.

Tsukamoto, Y. (1979). An approach to fuzzy reasoning method. *Advances in Fuzzy Set Theory and Applications*, M. M. Gupta, R. K. Ragade, and R. R. Yager, Eds., Amsterdam: North-Holland, 137-149.

Wang, L.X., & Mendel, J.M. (1992). Generating Fuzzy Rules by Learning from Examples. *IEEE Transactions on System, Man and Cybernetics*, **22**(6), 1414-1427.

Weigend, A.S., Huberman, B.A., & Rumelhart, D.E. (1990). Predicting the future: A connectionist approach. *International Journal of Neural Systems*, **1**, 193-209.

Yamakawa, T., Uchino, E., Miki, T., & Kusanagi, H. (1992). A Neo Fuzzy Neuron and Its Application to System Identification and prediction of the System Behaviour. *Proceedings of the 2nd International Conference on Fuzzy Logic and Neural Networks*, Iizuka, Japan, 477-483.

Zadeh, L.A. (1965). Fuzzy Sets. *Information and Control*, **8**, 338-353.

Zadeh, L.A. (1994). Fuzzy Logic, Neural Networks, and Soft Computing. *Communications of the ACM*, **37**(3), 77-84.

Table Legends

Fuzzy rules	**IF**		**THEN**	Degree
	x_1 *is*	x_2 *is*	y *is*	
1	VS	L	VL	0.57
2	VS	VL	VL	0.88
3	S	M	M	0.93
4	S	L	L	1.00
5	S	VL	VL	0.77
6	M	S	S	0.99
7	M	M	M	0.99
8	M	L	L	0.93
9*	M	VL	VL	0.23
10	L	VS	VS	0.67
11	L	S	S	0.98
12	L	M	M	0.69
13*	L	L	L	0.22
14	VL	VS	VS	0.98
15	VL	S	S	0.63

Table 1: The fuzzy rule base generated from the desired 292 input-output data pairs of Box & Jenkins Data and their corresponding degrees.

Fuzzy rules	**IF**				**THEN**	Degree
	x_1 *is*	x_2 *is*	x_3 *is*	x_4 *is*	y *is*	
1	S	S	S	S	M	0.56
2	S	S	S	M	M	0.46
3	S	M	M	M	S	0.22
4	S	M	M	L	S	0.32
5	M	S	S	S	M	0.59
6	M	S	S	M	M	0.21
7	M	M	S	S	M	0.46
8	M	M	M	S	M	0.44
9	M	M	M	M	M	0.40
10	M	M	M	L	M	0.33
11	M	L	L	M	M	0.12
12	M	L	L	L	M	0.55
13	L	M	M	S	L	0.16
14	L	M	M	M	L	0.21
15	L	L	L	M	L	0.31
16	L	L	L	L	M	0.49

Table 2: The fuzzy rules generated from the desired input-output pairs of Mackey & Glass and their corresponding degrees.

Fuzzy rules	**IF**				**THEN**	Degree
	x_1 *is*	x_2 *is*	x_3 *is*	x_4 *is*	y *is*	
1	S	S	S	S	M	0.77
2*	S	S	S	M	S^*	0.69
3	S	M	M	M	S	0.37
4	S	M	M	L	S	0.47
5	M	S	S	S	M	0.82
6	M	S	S	M	M	0.29
7	M	M	S	S	M	0.71
8	M	M	M	S	M	0.66
9	M	M	M	M	M	0.63
10	M	M	M	L	M	0.47
11	M	L	L	M	M	0.18
12	M	L	L	L	M	0.83
13	L	M	M	S	L	0.20
14	L	M	M	M	L	0.33
15	L	L	L	M	L	0.44
16	L	L	L	L	M	0.75

Table 3: The fuzzy rules generated using the adaptive procedure of Mackey & Glass time series and their corresponding degrees.

Figure Legends

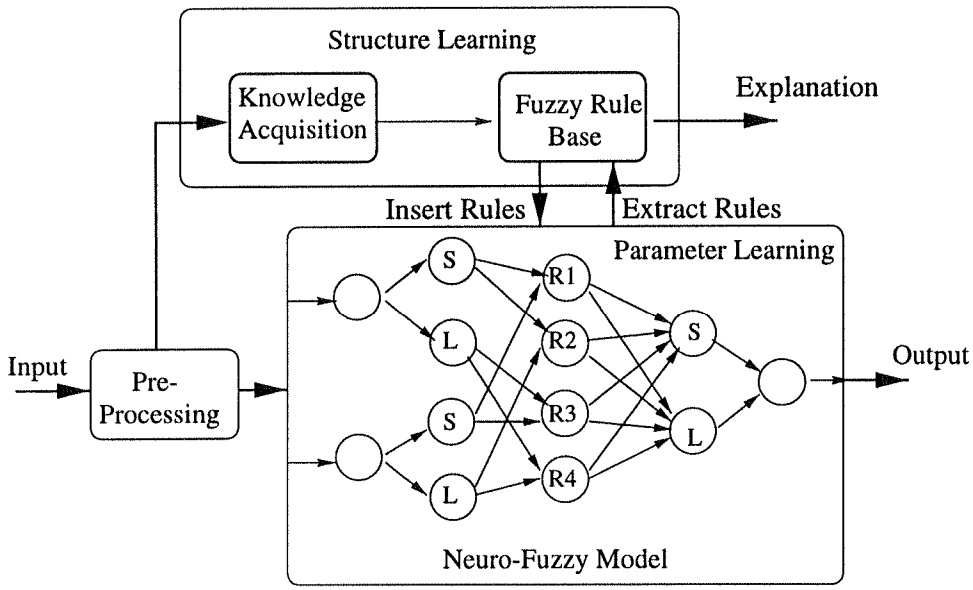

Figure 1: A general schematic diagram of the HyFIS.

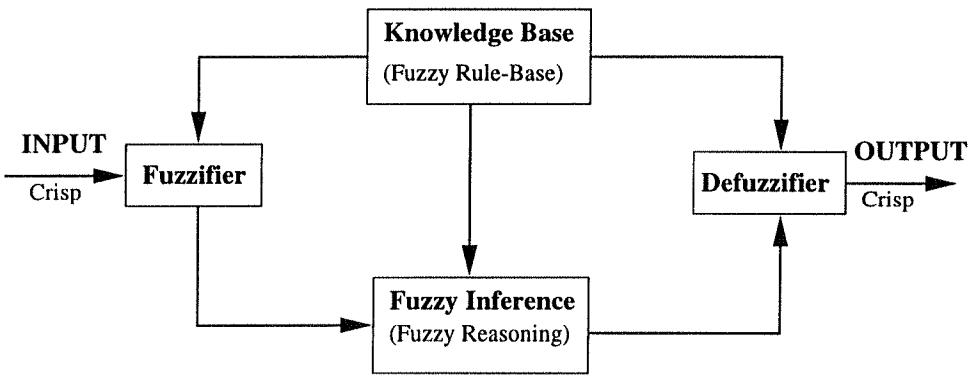

Figure 2: The basic structure of a fuzzy inference system (FIS).

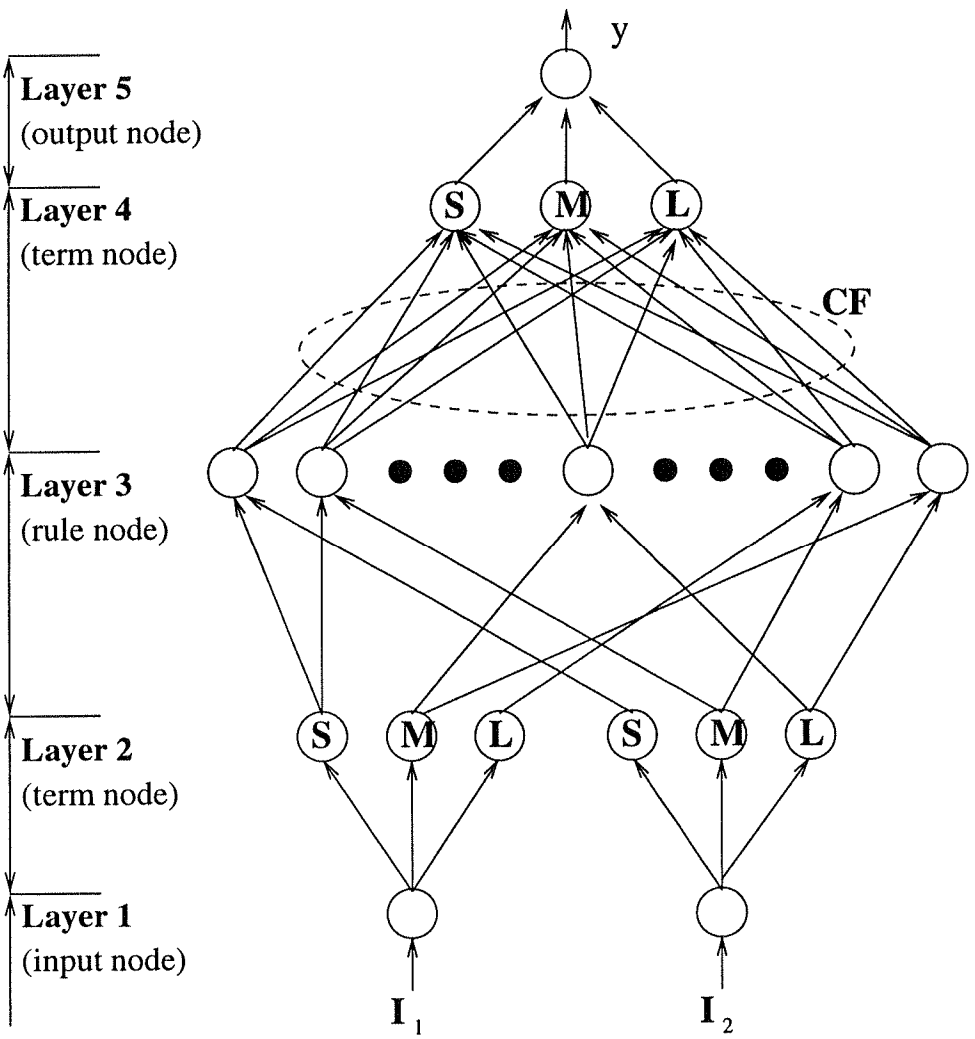

Figure 3: The structure of the Neuro-Fuzzy model from the HyFIS architecture.

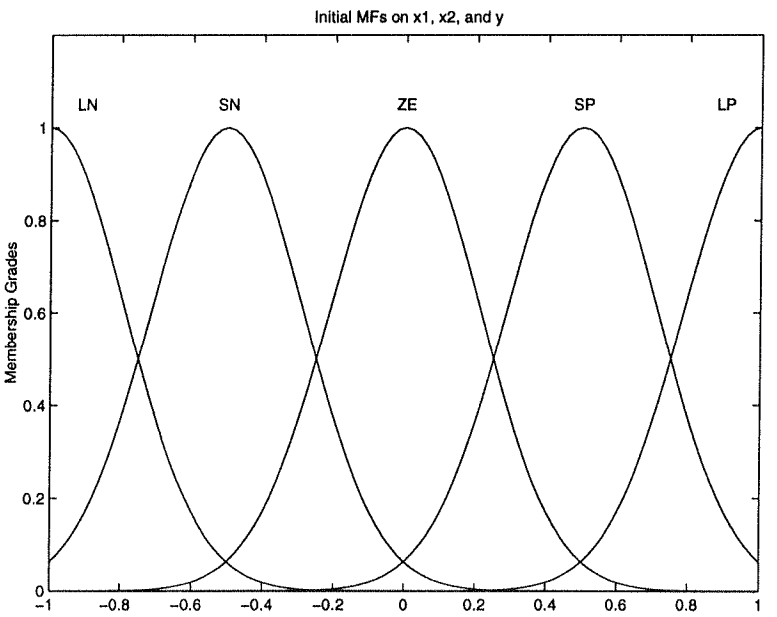

Figure 4: Initial division of input and output spaces into five fuzzy regions and the corresponding Gaussian membership functions.

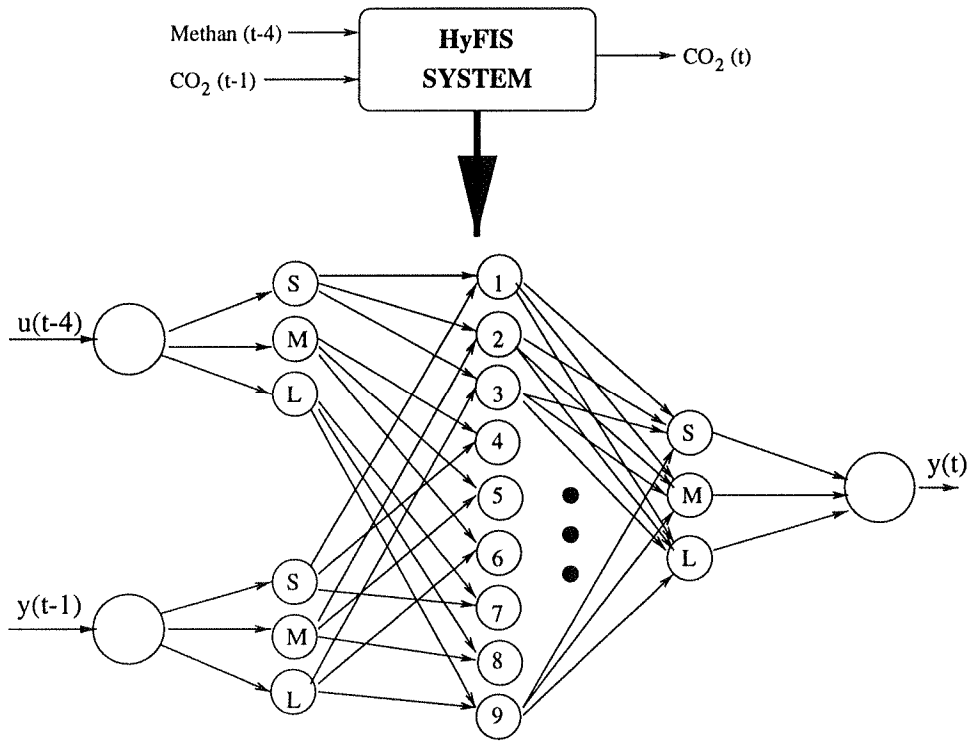

Figure 5: A neuro-fuzzy model for the Box-Jenkins data set.

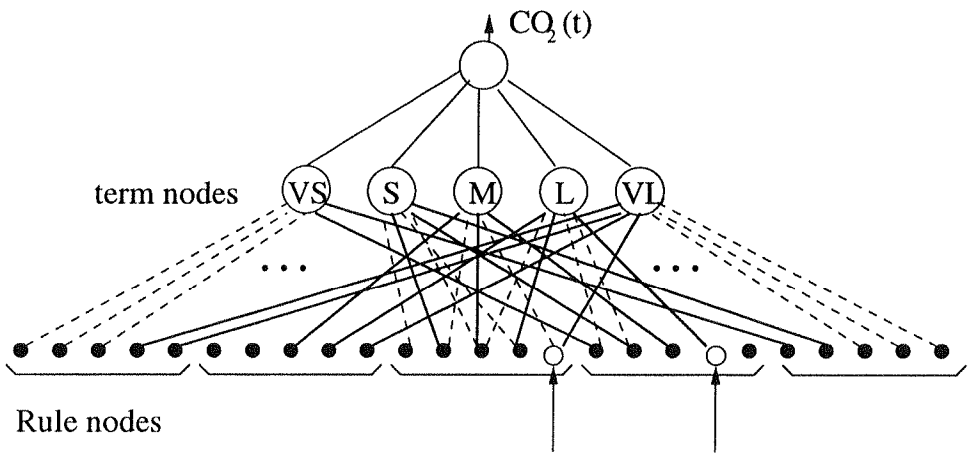

Figure 6: Initialised structure of the neuro-fuzzy model for the Box-Jenkins data set when 5 MFs are used and 13 rules are inserted; later two more rules (the *empty* circle nodes) are added to the structure.

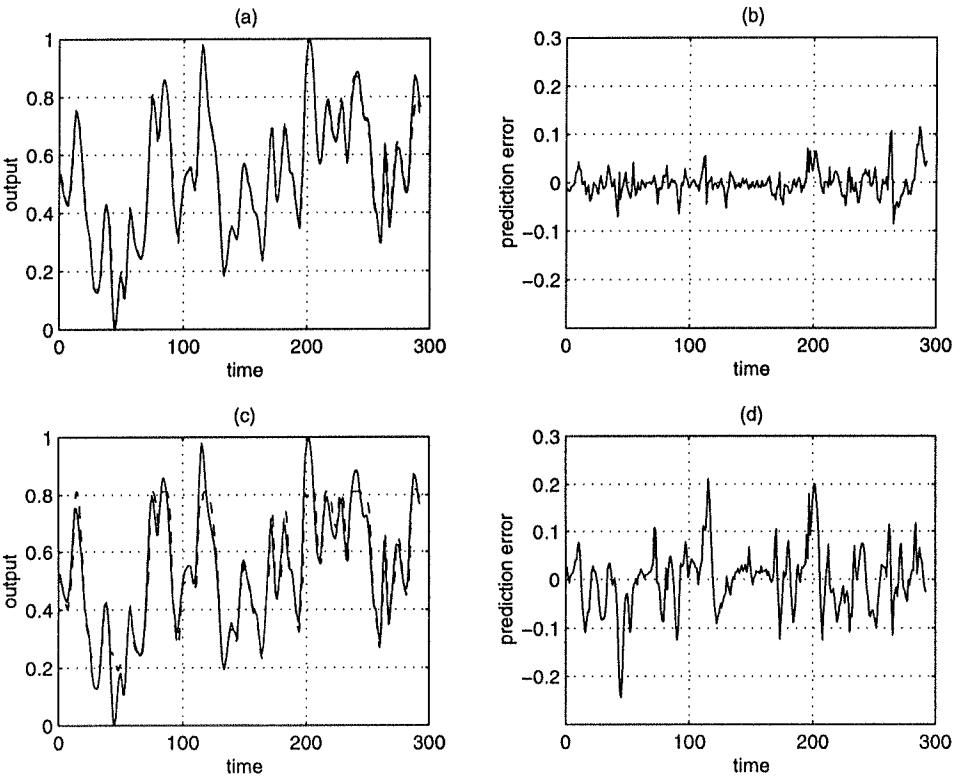

Figure 7: HyFIS for the Box-Jenkins data for the first and the second cases: (a) the desired system response (solid line) and the HyFIS prediction (dashed line); (b) the HyFIS prediction error for using fuzzy rules from the partitioned data pairs (200 pairs) only; (c) the desired system response (solid line) and the HyFIS prediction (dashed line); (d) the HyFIS prediction error for a smaller set of fuzzy rules (92 pairs only).

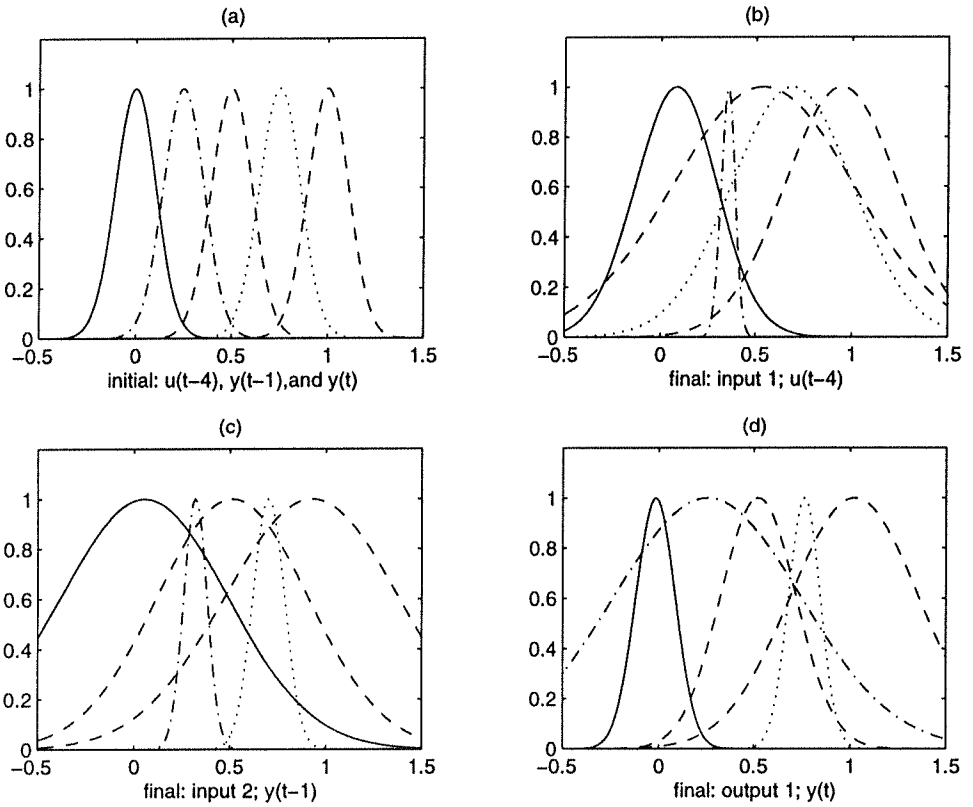

Figure 8: Initial and learned membership functions for the fuzzy rules from the first 200 examples of the Box & Jenkins Data Set.

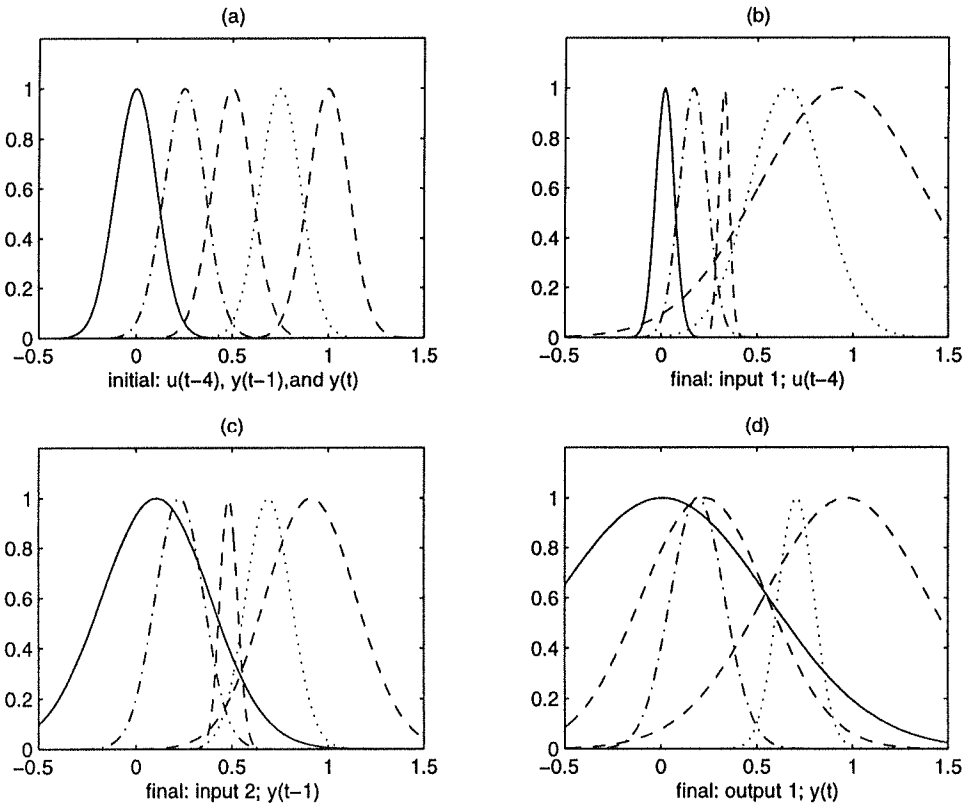

Figure 9: Initial and learned membership functions for a smaller set of rules (from 92 pairs) for the Box & Jenkins Data Set.

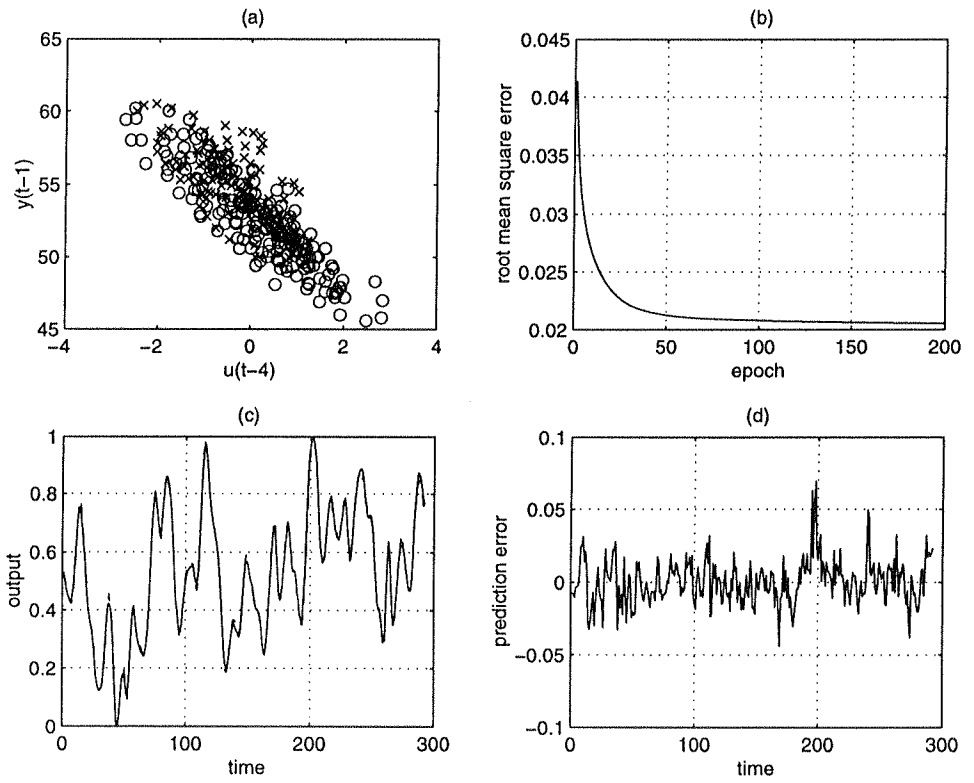

Figure 10: An HyFIS for the Box & Jenkins data for a combination of the first and second case: (a) training and test data distribution; (b) training error curve; (c) desired system response (solid) and HyFIS prediction (dashed); (d) HyFIS prediction error.

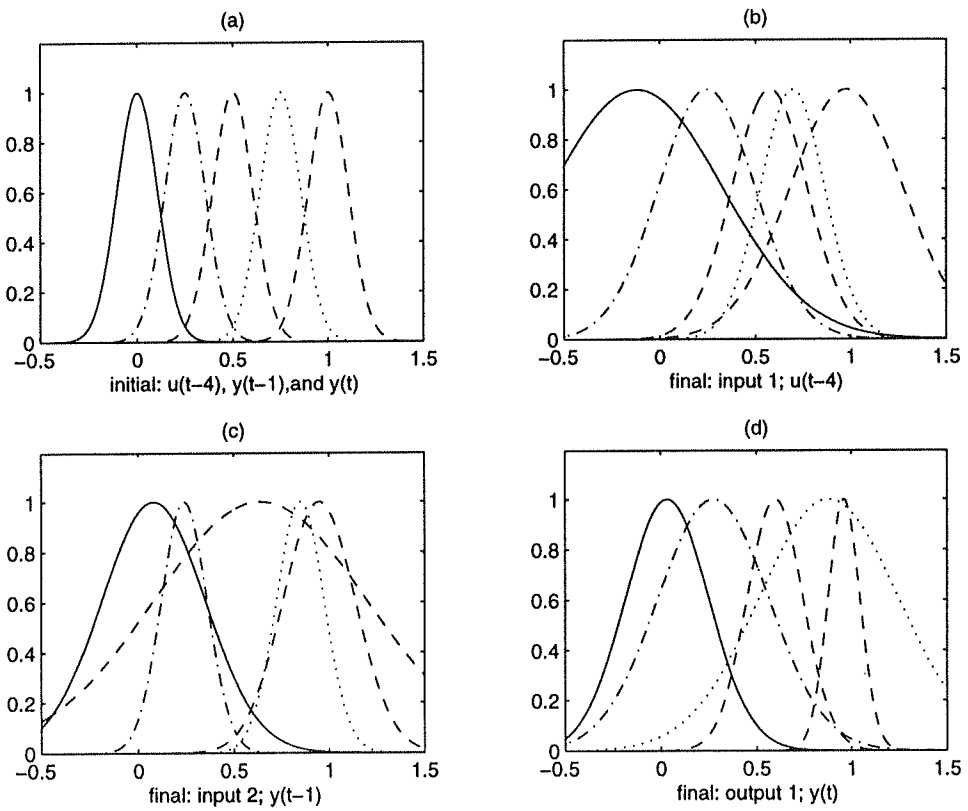

Figure 11: The initial and the learned membership functions for a combination of linguistic fuzzy rules and fuzzy rules generated from the desired input-output data pairs for Box & Jenkins Data Set.

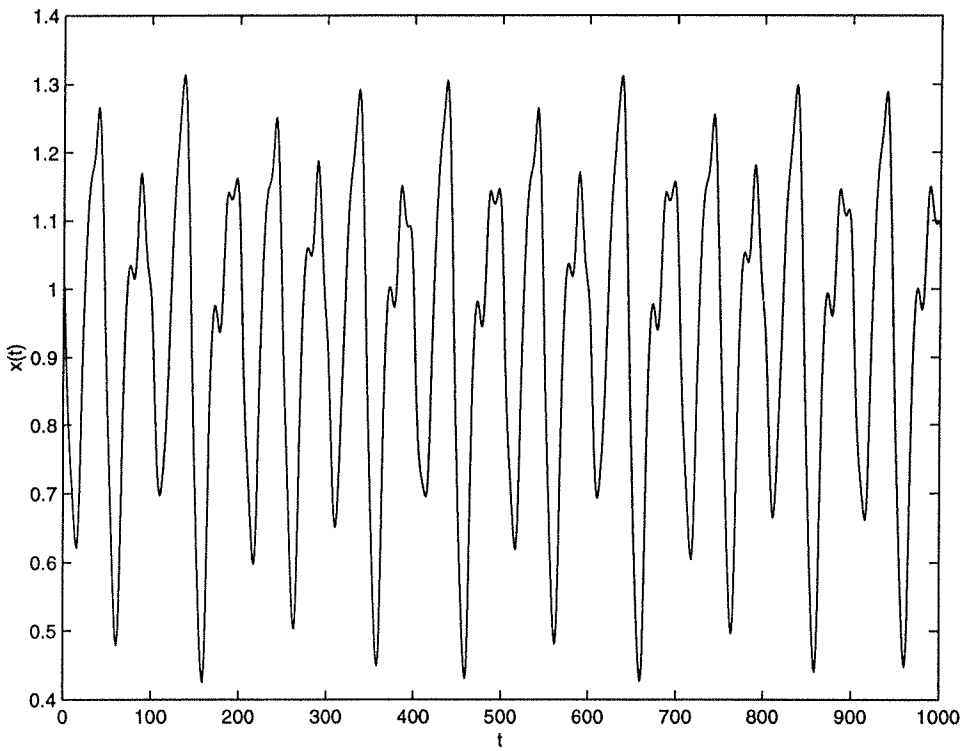

Figure 12: The Mackey-Glass time Series at $\tau = 17$ exhibits a chaotic behaviour

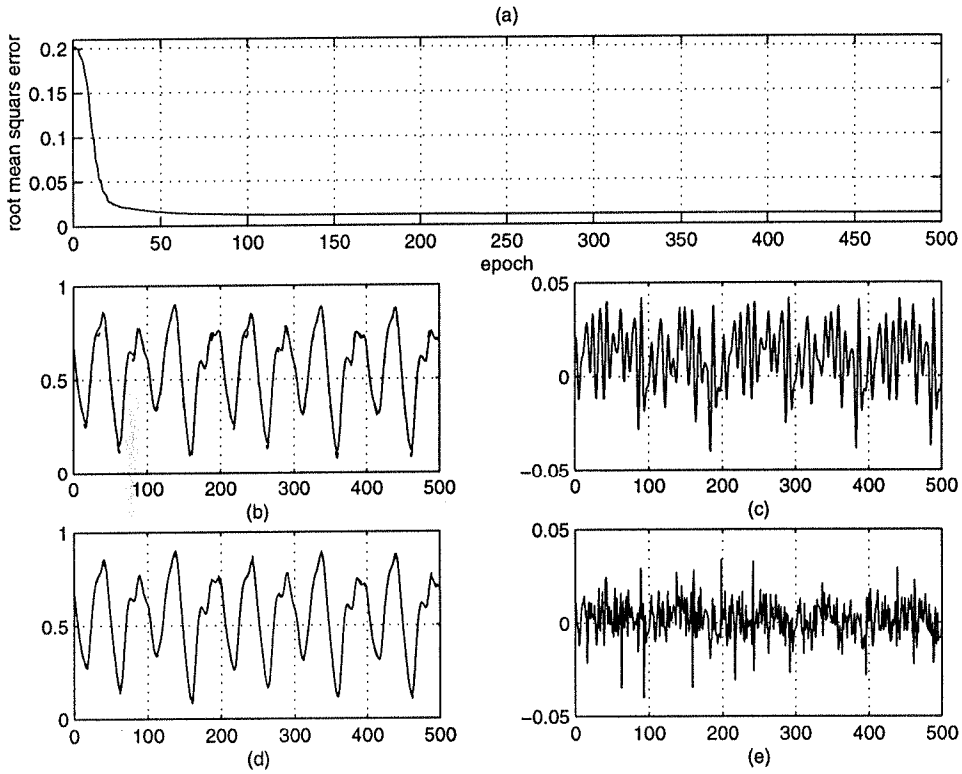

Figure 13: Generalisation test of the HyFIS model for the Mackey & Glass time-series: (a) training RMSE curves for the model; (b) the desired (solid) and predicted (dashed) time series; (c) prediction error from 501 to 1000 when 500 training data are used; (d) the desired (solid) and predicted (dashed) time series; (e) prediction error for data items from 501 to 1000 using adapted fuzzy rule base.

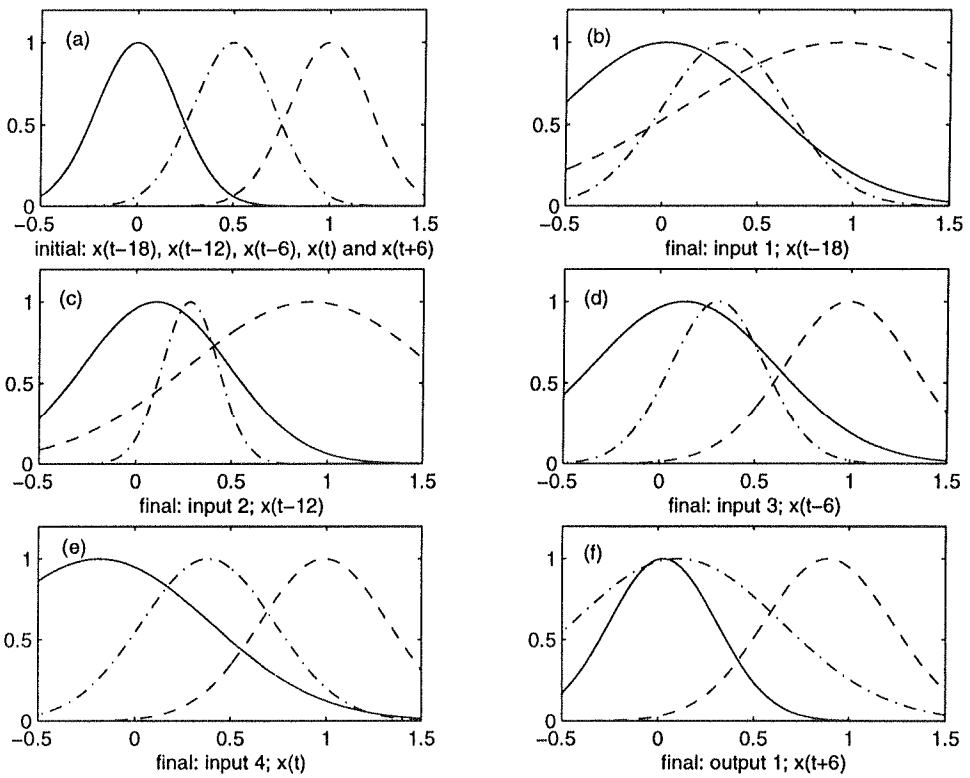

Figure 14: Initial and learned membership functions for the input and the output variables for the Mackey & Glass time-series.