Chapter Number

Adaptive Neuro-Fuzzy Systems

Azar, Ahmad Taher
Electrical Communication & Electronics Systems Engineering department, Modern Science and Arts University (MSA), 6th of October City, Egypt

1. Introduction

One benefit of fuzzy systems (Zadeh, 1965; Ruspini et al., 1998; Cox, 1994) is that the rule base can be created from expert knowledge, used to specify fuzzy sets to partition all variables and a sufficient number of fuzzy rules to describe the input/output relation of the problem at hand. However, a fuzzy system that is constructed by expert knowledge alone will usually not perform as required when it is applied because the expert can be wrong about the location of the fuzzy sets and the number of rules. A manual tuning process must usually be appended to the design stage which results in modifying the membership functions and/or the rule base of the fuzzy system. This tuning process can be very time-consuming and error-prone. Also, in many applications expert knowledge is only partially available or not at all. It is therefore useful to support the definition of the fuzzy rule base by automatic learning approaches that make use of available data samples. This is possible since, once the components of the fuzzy system is put in a parametric form, the fuzzy inference system becomes a parametric model which can be tuned by a learning procedure. Fuzzy logic and artificial neural networks (Haykin, 1998; Mehrotra et al., 1997) are complementary technologies in the design of intelligent systems. The combination of these two technologies into an integrated system appears to be a promising path toward the development of Intelligent Systems capable of capturing qualities characterizing the human brain. Both neural networks and fuzzy logic are powerful design techniques that have their strengths and weaknesses. Table 1 shows a comparison of the properties of these two technologies (Fuller, 2000). The integrated system will have the advantages of both neural networks (e.g. learning abilities, optimization abilities and connectionist structures) and fuzzy systems (humanlike IF-THEN rules thinking and ease of incorporating expert knowledge) (Brown & Harris,1994). In this way, it is possible to bring the low-level learning and computational power of neural networks into fuzzy systems and also high-level humanlike IF-THEN thinking and reasoning of fuzzy systems into neural networks. Thus, on the neural side, more and more transparency is pursued and obtained either by pre-structuring a neural network to improve its performance or by possible interpretation of the weight matrix following the learning stage. On the fuzzy side, the development of methods allowing automatic tuning of the parameters that characterize the fuzzy system can largely draw inspiration from similar methods used in the connectionist community. This combination does not usually mean that a neural network and a fuzzy system are used together in some way.

The neuro-fuzzy method is rather a way to create a fuzzy model from data by some kind of learning method that is motivated by learning procedures used in neural networks. This substantially reduces development time and cost while improving the accuracy of the resulting fuzzy model. Being able to utilize a neural learning algorithm implies that a fuzzy system with linguistic information in its rule base can be updated or adapted using numerical information to gain an even greater advantage over a neural network that cannot make use of linguistic information and behaves as a black-box. Equivalent terms for neuro-fuzzy systems that can be found in the literature are neural fuzzy or sometimes neuro-fuzzy networks (Buckley & Eslami, 1996). Neuro-fuzzy systems are basically adaptive fuzzy systems developed by exploiting the similarities between fuzzy systems and certain forms of neural networks, which fall in the class of generalized local methods. Hence, the behavior of a neuro-fuzzy system can either be represented by a set of humanly understandable rules or by a combination of localized basis functions associated with local models (i.e. a generalized local method), making them an ideal framework to perform nonlinear predictive modeling. Nevertheless, one important consequence of this hybridization between the representational aspect of fuzzy models and the learning mechanism of neural networks is the contrast between readability and performance of the resulting model.

Skills	Type	Fuzzy Systems	Neural Nets
Knowledge acquisition	Inputs	Human experts	Sample sets
	Tools	Interaction	Algorithms
Unceratinity	Information	Quantitative and Qualitative	Quantitative
Reasoning	Cognition	Heuristic approach	Perception
	Mechanism	Low	Parallel Computation
	Speed	low	High
Adaption	Fault-tolerance	Low	Very high
	Learning	Induction	Adjusting weights
Natural language	Implementation	Explicit	Implicit
	Flexibility	High	Low

Table 1. Properties of neural networks and fuzzy Systems (Fuller, 2000).

Summarizing, neural networks can improve their transparency, making them closer to fuzzy systems, while fuzzy systems can self-adapt, making them closer to neural networks (Lin & Lee, 1996). Fuzzy systems can be seen as a special case of local modeling methods, where the input space is partitioned into a number of fuzzy regions represented by multivariate membership functions. For each region, a rule is defined that specifies the output of the system in that region. The class of functions that can be accurately reproduced by the resulting model is determined by the nonlinear mapping performed by the multivariate fuzzy membership functions. This impressive result allows comparison to be drawn between fuzzy systems and the more conventional techniques referred to as generalized local methods. In particular, if bell-shaped (Gaussian) membership functions are used, then a Takagi-Sugeno (TS) fuzzy system is equivalent to a special kind of Radial Basis Function (RBF) network (Jang & Sun, 1993).

Theorems and analysis derived for local modeling methods can directly be applied to fuzzy systems. Also, due to this similarity, fuzzy systems allow relatively easy application of learning techniques used in local methods for identification of fuzzy rules from data. On the other side, fuzzy systems distinguish from other local modeling techniques, for their potentiality of an easy pre-structuring and a convenient integration of a priori knowledge. Many learning algorithms from the area of local modeling, and more specifically techniques developed for some kind of neural networks, have been extended to automatically extract or tune fuzzy rules based on available data. All these techniques exploit the fact that, at the computational level, a fuzzy system can be seen as a layered architecture, similar to an artificial neural network. By doing so, the fuzzy system becomes a neuro-fuzzy system, i.e. special neural network architecture. In 1991, Lin and Lee have proposed the very first implementation of Mamdani fuzzy models using layered feed-forward architecture (Lin & Lee, 1991). Nevertheless, the most famous example of neuro-fuzzy network is the Adaptive Network-based Fuzzy Inference System (ANFIS) developed by Jang in 1993 (Jang, 1993), that implements a TS fuzzy system in a network architecture, and applies a mixture of plain back-propagation and least mean squares procedure to train the system.

2. Types of neuro-fuzzy systems

There are several ways to combine neural networks and fuzzy logic. Efforts at merging these two technologies may be characterized by considering three main categories: neural fuzzy systems, fuzzy neural networks and fuzzy-neural hybrid systems.

2.1 Neural fuzzy systems

Neural fuzzy systems are characterized by the use of neural networks to provide fuzzy systems with a kind of automatic tuning method, but without altering their functionality. One example of this approach would be the use of neural networks for the membership function elicitation and mapping between fuzzy sets that are utilized as fuzzy rules as shown in Fig. 1. In the training process, a neural network adjusts its weights in order to minimize the mean square error between the output of the network and the desired output. In this particular example, the weights of the neural network represent the parameters of the fuzzification function, fuzzy word membership function, fuzzy rule confidences and

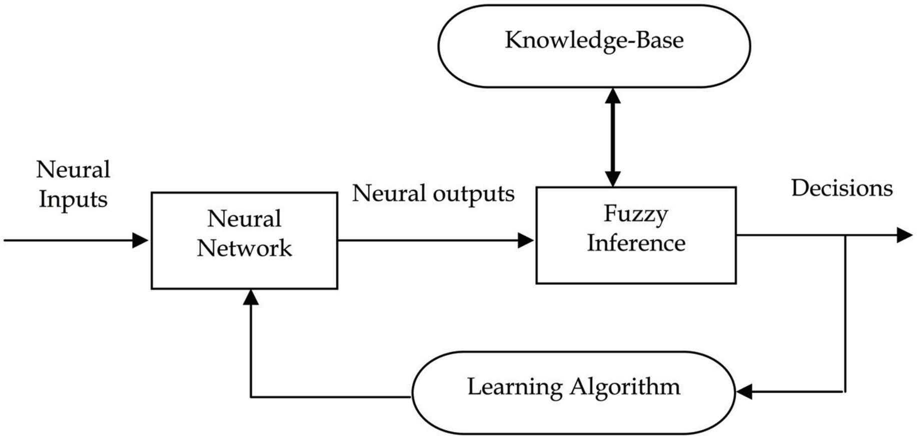

Fig. 1. Neural fuzzy system (Fuller, 2000).

defuzzification function respectively. In this sense, the training of this neural network results in automatically adjusting the parameters of a fuzzy system and finding their optimal values.This kind of combination is mostly used in control applications. Examples of this approach can be found in (Wang & Mendel, 1992; Nomura et al., 1992; Nauck, 1994; Shi & Mizumoto, 2000a; Shi & Mizumoto, 2000b; Yager & Filev, 1994; Cho & Wang, 1996; Ichihashi & Tuksen, 1993).

2.2 Fuzzy neural systems

The main goal of this approach is to 'fuzzify' some of the elements of neural networks, using fuzzy logic (Fig. 2). In this case, a crisp neuron can become fuzzy. Since fuzzy neural networks are inherently neural networks, they are mostly used in pattern recognition applications. In 1996, for instance, Lin and Lee presented a neural network composed of fuzzy neurons (Lin and Lee, 1996). In these fuzzy neurons, the inputs are non-fuzzy, but the weighting operations are replaced by membership functions. The result of each weighting operation is the membership value of the corresponding input in the fuzzy set. Also, the aggregation operation may use any aggregation operators such as min and max and any other t-norms and t-conorms.

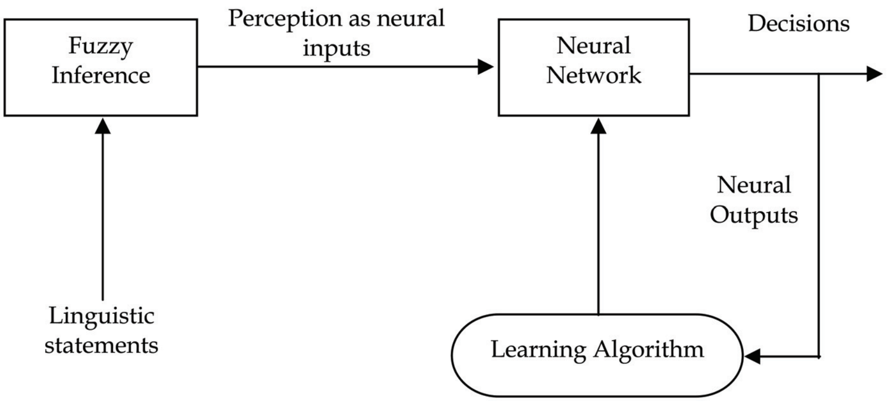

Fig. 2. Fuzzy neural system (Fuller, 2000).

2.3 Hybrid neuro-fuzzy systems

In this approach, both fuzzy and neural networks techniques are used independently, becoming, in this sense, a hybrid system. Each one does its own job in serving different functions in the system, incorporating and complementing each other in order to achieve a common goal. This kind of merging is application-oriented and suitable for both control and pattern recognition applications. The idea of a hybrid model is the interpretation of the fuzzy rule-base in terms of a neural network. In this way the fuzzy sets can be interpreted as weights, and the rules, input variables, and output variables can be represented as neurons. The learning algorithm results, like in neural networks, in a change of the architecture, i.e. in an adaption of the weights, and/or in creating or deleting connections. These changes can be interpreted both in terms of a neural net and in terms of a fuzzy controller. This last aspect is very important as the black box behaviour of neural nets is avoided this way. This means a successful learning procedure results in an explicit increase of knowledge that can be represented in form of a fuzzy controller's rule base. Hybrid neuro-fuzzy controllers are realized by approaches like ARIC (Berenji, 1992), GARIC (Bersini et al., 1993), ANFIS (Jang, 1993) or the NNDFR model (Takagi & Hayashi 1991). These approaches consist all more or less of special neural networks, and they are capable to learn fuzzy sets.

3. Adaptive-Neuro-Fuzzy Inference System: ANFIS

Adaptive Neuro-Fuzzy Inference System (ANFIS) is one of the most successful schemes which combine the benefits of these two powerful paradigms into a single capsule (Jang, 1993). An ANFIS works by applying neural learning rules to identify and tune the parameters and structure of a Fuzzy Inference System (FIS). There are several features of the ANFIS which enable it to achieve great success in a wide range of scientific applications. The attractive features of an ANFIS include: easy to implement, fast and accurate learning, strong generalization abilities, excellent explanation facilities through fuzzy rules, and easy to incorporate both linguistic and numeric knowledge for problem solving (Jang & Sun, 1995; Jang et al., 1997). According to the neuro-fuzzy approach, a neural network is proposed to implement the fuzzy system, so that structure and parameter identification of the fuzzy rule base are accomplished by defining, adapting and optimizing the topology and the parameters of the corresponding neuro-fuzzy network, based only on the available data. The network can be regarded both as an adaptive fuzzy inference system with the capability of learning fuzzy rules from data, and as a connectionist architecture provided with linguistic meaning. A typical architecture of an ANFIS, in which a circle indicates a fixed node, whereas a square indicates an adaptive node, is shown in Figure 3. In this

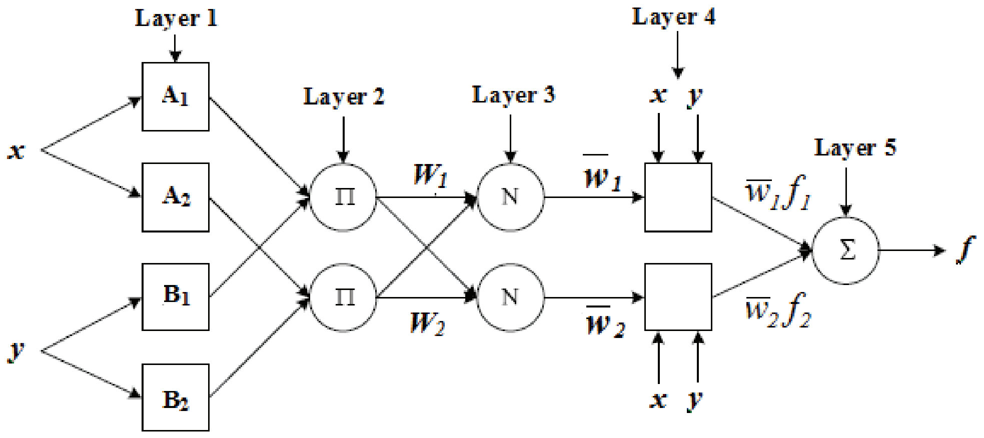

Fig. 3. First order Sugeno ANFIS architecture (Type-3 ANFIS) (Jang, 1993).

connectionist structure, there are input and output nodes, and in the hidden layers, there are nodes functioning as membership functions (MFs) and rules. This eliminates the disadvantage of a normal feedforward multilayer network, which is difficult for an observer to understand or to modify. For simplicity, we assume that the examined FIS has two inputs and one output. For a first-order Sugeno fuzzy model, a typical rule set with two fuzzy "if-then" rules can be expressed as follows:

Rule 1: If x is A_1 and y is B_1, then $f_1 = p_1 x + q_1 y + r_1$
Rule 2: If x is A_2 and y is B_2, then $f_2 = p_1 x + q_2 y + r_2$

Where x and y are the two crisp inputs, and A_i and B_i are the linguistic labels associated with the node function.
As indicated in Fig. 3, the system has a total of five layers. The functioning of each layer is described as follows (Jang, 1993).
Input node (Layer 1): Nodes in this layer contains membership functions. Parameters in this layer are referred to as premise parameters. Every node *i* in this layer is a square and adaptive node with a node function:

$$O_i^1 = \mu_{A_i}(x) \qquad \text{For i = 1, 2} \tag{1}$$

Where x is the input to node *i*, and A_i is the linguistic label (small , large, etc.) associated with this node function. In other words, O_i^1 is the membership function of A_i and it specifies the degree to which the given x satisfies the quantifier A_i.
Rule nodes (Layer 2): Every node in this layer is a circle node labeled II, whose output represents a firing strength of a rule. This layer chooses the minimum value of two input weights. In this layer, the AND/OR operator is applied to get one output that represents the results of the antecedent for a fuzzy rule, that is, firing strength. It means the degrees by which the antecedent part of the rule is satisfied and it indicates the shape of the output function for that rule. The node generates the output (firing strength) by cross multiplying all the incoming signals:

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

Average nodes (Layer 3): Every node in this layer is a circle node labeled N. The *i*th node calculates the ratio between the *i*th rule's firing strength to the sum of all rules' firing strengths. Every node of these layers calculates the weight, which is normalized.
For convenience, outputs of this layer are called normalized firing strengths.

$$\overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{3}$$

Consequent nodes (Layer 4): This layer includes linear functions, which are functions of the input signals. This means that the contribution of ith rule's towards the total output or the model output and/or the function defined is calculated. Every node *i* in this layer is a square node with a node function:

$$O_i^4 = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i) \tag{4}$$

Where $\overline{w}_i$ is the output of layer 3, and $\{p_i, q_i, r_i\}$ is the parameter set of this node. These parameters are referred to as consequent parameters.
***Output node (Layer 5)*:** The single node in this layer is a fixed node labeled $\sum$, which computes the overall output by summing all incoming signals:

$$O_i^5 = \text{overalloutput} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{5}$$

4. Modeling with neuro-fuzzy systems

Whatever may be the adopted vision of fuzzy model, two different phases must be carried out in fuzzy modeling, designated as structural and parametric identification. Structural identification consists of determining the structure of the rules, i.e. the number of rules and the number of fuzzy sets used to partition each variable in the input and output space so as to derive linguistic labels. Once a satisfactory structure is available, the parametric identification must follow for the fine adjustment of the position of all membership functions together with their shape as the main concern. As seen before, to overcome the limitations of using expert knowledge in defining the fuzzy rules, data driven methods to create fuzzy systems are needed. With such methods both structure and parameters are derived from scratch relying only on the training data. There are several ways that structure learning and parameter learning can be combined in a neuro-fuzzy system. They can be performed sequentially: structure learning is used first to find an appropriate structure of the fuzzy rule base, and then parameter learning is used to identify the parameters of each rule. In some neuro-fuzzy systems the structure is fixed and only parameter learning is performed. Algorithms inspired by neural network learning often do parameter learning. Structure learning on the other hand is usually not from neural networks. Indeed, many different approaches exist to automatically determine the structure of neural networks, but none of them is appropriate to perform structure identification in neuro-fuzzy models. In the following, different methods are presented that used for structure and parameter identification in neuro-fuzzy systems. There may be a lot of structure/parameter combinations which make the fuzzy model to behave satisfactorily; hence the search for the best model is not an easy task.
As a rule, simple fuzzy models should be preferred to complex ones; hence in the search for the best model two main objectives must be taken into account: good accuracy and minimal complexity.

4.1 Parametric identification

Two types of parameters characterize a fuzzy model: those determining the shape and distribution of the input fuzzy sets and those describing the output fuzzy sets (or linear models). Many neuro-fuzzy systems use direct nonlinear optimization to identify all the parameters of a fuzzy system. Different optimization techniques can be used to this aim. The most widely used is an extension of the well-known back-propagation algorithm implemented by gradient descent. A very large number of neuro-fuzzy systems are based on backpropagation. One limitation of using gradient descent techniques is that the membership functions and all functions that take part in the inference of the fuzzy rule base must be differentiable. As a consequence, gradient descent learning can be more easily applied to identify the parameters of a TS model, because only the product operator is used for intersection and the output is computed as a weighted sum. Recent neuro-fuzzy approaches choose to implement back-propagation by simple heuristics instead of gradient descent to identify the parameters of a Mamdani-type fuzzy model (Nauck & Kruse, 1999).

The general idea of such heuristics is to slightly modify the membership functions of a fuzzy rule according to how much the rule contributes to the overall output of the fuzzy system. From the proposed type-3 ANFIS architecture (see Fig. 3), it is observed that given the values of premise parameters, the overall output can be expressed as a linear combinations of the consequent parameters. More precisely, the output f in Fig. 3 can be rewritten as:

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 = \overline{w}_1 f_1 + \overline{w}_2 f_2 \\ &= (\overline{w}_1 x) p_1 + (\overline{w}_1 y) q_1 + (\overline{w}_1) r_1 + (\overline{w}_2 x) p_2 + (\overline{w}_2 y) q_2 + (\overline{w}_2) r_2 \end{aligned} \tag{6}$$

Which is linear in the consequent parameters (p_1, q_1, r_1, p_2, q_2 and r_2). Therefore the hybrid learning algorithm can be applied directly. More specifically, in the forward pass of the hybrid learning algorithm, functional signals go forward till layer 4 and the consequent parameters are identified by the least squares estimate (LSE). In the backward pass, the error rates propagate backward and the premise parameters are updated by the gradient descent. Table 2 summarizes the activities in each pass. As mentioned earlier, the consequent parameters thus identified are optimal (in the consequent parameter space) under the condition that the premise parameters are fixed.

	Forward Pass	**Backward pass**
Premise Parameters	Fixed	Gradient Descent
Consequent parameters	Least-squares estimator	Fixed
Signals	Node Outputs	Error Signals

Table 2 The two passes in the hybrid learning algorithm (Jang & Sun, 1995).

However, it should be noted that the computation complexity of the least squares estimate is higher than that of the gradient descent. In fact, there are four methods to update the parameters, as listed below according to their computation complexities (Jang, 1993):

- Gradient Descent Only: All parameters are updated by the gradient descent.
- Gradient Descent and One Pass of LSE: The LSE is applied only once at the beginning to get the initial value of the consequent parameters and then the gradient descent takes over to update all parameters.
- Gradient descent and LSE: This is the proposed hybrid learning rule.
- Sequential (Approximate) LSE Only: The ANFIS is linearized with respect to the premise parameters and the extended Kalman filter algorithm is employed to update all parameters.

The choice of above methods should be based on the trade-off between computation complexity and resulting performance. Other approaches to parameter learning of fuzzy models that do not require gradient computations, and hence differentiability, are reinforcement learning which requires only a single scalar evaluation of the output, and Genetic Algorithms (GAs) that perform a random search in the parameter space, using a population of individuals, each coding the parameters of a potential fuzzy rule base (Seng et al., 1999). One problem with GAs is that with conventional binary coding, the length of individuals increases significantly with the number of inputs, the number of fuzzy sets and the number of rules. Evolution Strategies (ES) are more suitable techniques to tune the fuzzy rule parameters due to their direct coding scheme (Jin et al, 1999). GA's and ES allow also a simultaneous identification of the parameters and the structure (rule number) of a fuzzy model, but in such a case these evolutionary techniques are computationally demanding since very complex individuals need to be manipulated. The identification of the whole set of parameters by nonlinear optimization techniques may be computationally intensive and requiring long convergence rates. To speed up the process of parameter identification, many neuro-fuzzy systems adopt a multi-stage learning procedure to find and optimize the parameters. Typically, two stages are considered. In the first stage the input space is partitioned into regions by unsupervised learning, and from each region the premise (and eventually the consequent) parameters of a fuzzy rule are derived. In the second stage the consequent parameters are estimated via a supervised learning technique. In most cases, the second stage performs also a fine adjustment of the premise parameters obtained in the first stage using a nonlinear optimization technique.

4.2 Structural identification

Before fuzzy rule parameters can be optimized, the structure of the fuzzy rule base must be defined. This involves determining the number of rules and the granularity of the data space, i.e. the number of fuzzy sets used to partition each variable. In fuzzy rule-based systems, as in any other modeling technique, there is a tradeoff between accuracy and complexity. The more rules, the finer the approximation of the nonlinear mapping can be obtained by the fuzzy system, but also more parameters have to be estimated, thus the cost and complexity increase. A possible approach to structure identification is to perform a stepwise search through the fuzzy model space. Once again, these search strategies fall into one of two general categories: forward selection and backward elimination.

- Forward selection. Starting from a very simple rule base, new fuzzy rules are dynamically added or the density of fuzzy sets is incrementally increased (Royas et al., 2000).
- Backward elimination. An initial fuzzy rule base, constructed from a priori knowledge or by learning from data, is reduced, until a minimum of the error function is found (Yen & Wang, 1999). The structure of the fuzzy rules can also be optimized by GA's so that a compact fuzzy rule base can be obtained (Seng et al., 1999).

The learning algorithm is an example of structure adaptation in neuro-fuzzy systems. Rules are dynamically recruited or deleted according to their significance to system performance, so that a parsimonious structure with high performance is achieved. When initial fuzzy rules are generated by clustering, the number of cluster (i.e. of rules) must be specified before clustering. If no prior knowledge is available that suggests the number of clusters, automated procedures can be applied. For example the number of clusters can be found by evaluating a given validity measure, i.e. a criterion that assesses the quality of the clusters, and selecting the number of clusters that minimizes (maximizes) the validity measure. Another approach is cluster merging, that starts with a high number of clusters and reduces them successively by merging compatible clusters until some threshold is reached and no more clusters can be merged.

5. Interpretability versus accuracy of neuro-fuzzy models

As seen in the previous sections, neuro-fuzzy systems are essentially fuzzy systems endowed with learning capabilities inspired (not only) by neural networks. Fuzzy systems join the advantages of modeling methods oriented to provide suitable models for both prediction and understanding. It must be considered whether these advantages of fuzzy systems for predictive modeling are preserved when they are transformed into neuro-fuzzy systems. The twofold face of fuzzy systems leads to a trade-off between readability and accuracy (table 3). Fuzzy systems can be forced to arbitrary precision, but it then loose interpretability. To be very precise, a fuzzy system needs a fine granularity and many fuzzy rules. It is obvious that the larger the rule base of a fuzzy system becomes, the less interpretable it gets (Nauck & Kruse, 1998a; Nauck & Kruse, 1998b).

	Interpretability	**Accuracy**
No. of parameters	Few Parameters	More Parameters
No. of fuzzy rules	Few Rules	More Rules
Type of Fuzzy logic Model	Mamdani Models	TSK models

Table 3. Interpretability vs. accuracy in fuzzy systems.

To keep the model simple, the prediction is usually less accurate. In solving this trade-off the interpretability (meaning also simplicity) of fuzzy systems must be considered the major advantage and hence it should be pursuit more than accuracy.

In fact fuzzy systems are not better function approximators or classifiers than other approaches. If we are interested in a very precise prediction, then we are usually not so much interested in the interpretability of the solution. In this case we use just one feature of fuzzy systems: the convenient combination of local models to an overall solution. For this, Sugeno-type models are more suited than Mamdani-type models because they offer more flexibility in the consequents of the rules. However, if optimal performance is the main objective, we should consider whether a fuzzy system is the most suitable approach and an exhaustive and deep comparison with related methods (local methods and generalized local methods) has to be done, in terms of pure performance, computational cost and practicability. Briefly put, fuzzy systems should be used for predictive modeling if an interpretable model is needed that can also be used to some extent for prediction. Interpretability of a fuzzy model should not mean that there is an exact match between the linguistic description of the model and the model parameters. This is not possible anyway, due to the subjective nature of fuzzy sets and linguistic terms. Usually it is not important that, for example, the term approximately zero be represented by a symmetrical triangular fuzzy set with support [-1, 1]. Interpretability means that the users of the model can accept the representation of the linguistic terms, more or less. The representation must roughly correspond to their intuitive understanding of the linguistic terms. Furthermore, interpretability should not mean that anybody could understand a fuzzy model. It means that users who are at least to some degree experts in the domain where the predictive modeling takes place can understand the model. Since interpretability itself is a fuzzy and subjective concept, it is hard to find an explicit and exhaustive list of conditions which, when violated, make the fuzzy model to lose its readability.

Traditional neuro-fuzzy modeling techniques, and in general data-driven methods for learning fuzzy rules from data, are aimed to optimize the prediction accuracy of the fuzzy model. However, while the accuracy improves, the transparency of the fuzzy models after learning may be lost. The overlap of the membership functions typically increases and peculiar situations may occur, when some membership functions are contained in the others or membership functions swap their positions. This hampers the interpretability of the final model. For the sake of interpretability, the learning procedure should take the semantics of the desired fuzzy system into account, and adhere to certain constraints, so that it cannot apply all the possible modifications to the parameters of a fuzzy system. For example the learning algorithms should be constrained such that adjacent membership functions do not exchange positions, do not move from positive to negative parts of the domains or vice versa, have a certain degree of overlapping, etc. The other important requirement to obtain interpretability is to keep the rule base small. A fuzzy model with interpretable membership functions but a very large number of rules is far from being understandable. By reducing the complexity, i.e. the number of parameters, of a fuzzy model, not only the rule base is kept manageable (hence the inference process is computationally cheaper) but also it can provide a more readable description of the process underlying the data. Also the use of a simple rule base contributes to decrease the overfitting, thus improving generalization. So far, few data-driven fuzzy rule learning methods aiming at improving the interpretability of the fuzzy models in terms of both small rule base and readable fuzzy sets have been proposed.

6. Case study: adaptive-neuro-fuzzy inference system as a novel approach for post-dialysis urea rebound prediction

6.1 Problem statement

Kinetic models of urea concentration are now widely used to manage hemodialysis (HD) patients. The calculation Kt/V (where K is the dialyzer clearance, t is the time of treatment, and V is the urea distribution volume), is now widely used to quantify HD treatment (Depner, 1994; Depner 1999). The Kt/V calculation is commonly determined from measurements of the pre-and post-HD blood urea nitrogen (BUN) concentrations (Gotch & Sargent,1985). However, because the rapid removal of BUN during HD causes a concentration disequilibrium between intracellular and extracellular fluid spaces, BUN increases immediately following HD. This phenomenon is well known as the urea rebound, and is due to the multiple-pool nature of the human body, and mass transfer resistance of the biological membranes and variations in regional blood flows (Schneditz & Daugirdas, 2001), Yashiro et al., 2004). Since Kt/V calculation is based in part on the post-hemodialysis BUN level, urea rebound has a significant impact upon the calculation of the delivered dose of hemodialysis. While single-pool kinetic modeling ($_{sp}$Kt/V) uses a convenient 30-second post-dialysis BUN sample, it does not take urea rebound into account, which leads to a 12 to 40% of the true equlilibrated dialysis dose ($_{eq}$Kt/V). Double-pool modeling ($_{eq}$Kt/V) uses an equilibrated BUN (C_{eq}) and is the best reflection of the true urea mass removed by hemodialysis. Because a delay of 30 to 60 minutes after dialysis before sampling the urea is inconvenient for both the clinician and patient, several methods have been devised to predict the PDUR in order to estimate the equilibrated Kt/V. The first is based on the standard single-pool Daugirdas Kt/V model that takes into account the dialysis time, which evolved into a double-pool Kt/V ($_{eq}$Kt/V) formula (Daugirdas & Schneditz, 1995). The second, according to Smye (Smye et al., 1994), Daugirdas (Daugirdas et al., 1996), Tattersall (Tattersall et al., 1996), and Maduell (Maduell et al., 1997), is based on an intradialytic urea sample at 33% of the session time. Other methods use a urea sample taken 30 minutes before the end of the hemodialysis session, which corresponds to the 30-minute PDUR (Bhaskaran et al., 1997, Canaud at al., 1997). Finally, Artificial Neural Network (ANN) method was used as a predictor of equilibrated post-dialysis blood urea concentration (C_{eq}) (Guh et al., 1998;

Azar et al., 2008a; Azar et al., 2009a). All of these methods still overestimate the urea rebound and underestimate the equlilibrated dialysis dose ($_{eq}$Kt/V).

6.2 Subjects and methods

The study was carried out at four dialysis centers. BUN was measured in all serum samples at a central laboratory. The overall study period was 5 months from August 1, 2008 to December 31, 2008. No subjects dropped out of the study. The study subjects consisted of 310 hemodialysis patients that gave their informed consent to participate. They are 165 male and 145 female patients, with ages ranging 14-75 years (48.97±12.77, mean and SD), and dialysis therapy duration ranging 6-138 months (50.56±34.67). The etiology of renal failure was chronic glomerulonephritis (65 patients), diabetic nephropathy (60 patients), vascular nephropathy (55 patients), hypertension (51 patients), interstitial chronic nephropathy (45 patients), other etiologies (18 patients) and unknown cause (16 patients). The vascular access was through a native arteriovenous fistula (285 patients), and a permanent jugular catheter (25 patients).

Patients had dialysis three times a week, in 3-4 hour sessions, with a pump arterial blood flow of 200-350 ml/min, and flow of the dialysis bath of 500-800 ml/min. The dialysate consisted of the following constituents: sodium 141 mmol/l, potassium 2.0 mmol/l, calcium 1.3 mmol/l, magnesium 0.2 mmol/l, chloride 108.0 mmol/l, acetate 3.0 mmol/l and bicarbonate 35.0 mmol/l. Special attention was paid to the real dialysis time, so that time-counters were fitted to all machines for all sessions, to record effective dialysis duration (excluding any unwanted interruptions, e.g. due to dialysis hypotensive episodes). All patients were dialyzed with 1.0 m^2 Polyethersulfone low flux dialyzer, 1.2 m^2 cellulose-synthetic low flux dialyzer (hemophane), 1.3 m^2 Polyethersulfone low flux dialyzer, 1.3 m^2 low flux polysulfone dialyzer, 1.6 m^2 low flux polysulfone dialyzer and 1.3 m^2 high flux polysulfone dialyzer. The dialysis technique was conventional hemodialysis, no patient being treated with hemodiafiltration. A Fresenius model 4008B and 4008S dialysis machine equipped with a volumetric ultrafiltration control system was used in each dialysis. Fluid removal was calculated as the difference between the patients' weight before dialysis and their target dry weight. Pre-dialysis body weight, blood pressure, pulse rate and axillary temperature were measured before ingestion of food and drink. Pre-dialysis BUN (C_{pre}) was sampled from the arterial port before the blood pump was started. Post-dialysis BUN (C_{post}) was obtained from the arterial port at the end of HD with the blood flow rate unchanged. Equilibrated post-dialysis BUN (C_{eq}) was obtained from the peripheral vein 30 and 60 minutes after HD. It was then corrected for urea generation. This corrected C_{eq} was used as a "gold standard" or the reference method.

6.3 ANFIS Architecture for equilibrated blood urea concentration prediction

To overcome the problem of overestimating urea rebound, Adaptive Neuro-Fuzzy Inference System (ANFIS) is developed in the form of a zero-order Takagi-Sugeno-Kang fuzzy inference system to predict equilibrated urea (C_{eq}) taken at 30 (C_{eq30}) and 60 (C_{eq60}) min after the end of the hemodialysis (HD) session in order to predict post dialysis urea rebound (PDUR) and equilibrated dialysis dose ($_{eq}$Kt/V) (Azar et al., 2008b; Azar, 2009b). The developed neuro-fuzzy hybrid approach is more accurate and doesn't require the model structure to be known a priori, in contrast to most of the modeling techniques. Also, this system doesn't require 30- or 60-minute post-dialysis urea sample. The proposed ANFIS can

construct an input-output mapping based on both expert knowledge (in the form of linguistic rules) and specified input-output data pairs and the least squares estimate (LSE) to identify the parameters (Jang et al., 1997). The ANFIS is a multilayer feed-forward network uses ANN learning algorithms and fuzzy reasoning to characterize an input space to an output space. The architecture of the proposed ANFIS realizes the inference mechanism of zero-order Takagi-Sugeno-Kang (TSK) fuzzy models (Takagi & Sugeno, 1985). The first-order Sugeno models have more freedom degrees and therefore the approximation ability is higher, together with a higher risk to overfit. The use of less freedom degrees is helping to control overfitting for the problem. Then, in this particular problem it is better zero-order. On the other hand, zero-order are more interpretable than first-order (depending on the number of rules required). Therefore, the selection of TSK model type depends on the necessities for the problem and the possibility to overfit the system (if it is important or not to have an interpretable model).
For an n-dimensional input, m-dimensional output fuzzy system, the rule base is composed of a set of fuzzy rules formally defined as:

$$R_k : IF\ (x_1\ is\ A_1^k) AND...AND\ (x_n\ is\ A_n^k)\ THEN\ (y_1\ is\ B_1^k)\ AND...AND\ (y_m\ is\ B_m^k)$$

Where x = (x_1, . . . x_n) are the input variables and $y = (y_1, \dots y_m)$ are the output variables, A_i^k are fuzzy sets defined on the input variables and B_j^k (j =1,…,m) are fuzzy singletons defined on the output variables over the output variables y_j. When y is constant, the resulting model is called "zero-order Sugeno fuzzy model", which can be viewed either as a special case of the Mamdani inference system (Mamdani & Assilian, 1975), in which each rule's consequent is specified by a fuzzy singleton, or a special case of the Tsukamoto fuzzy model (Tsukamoto, 1979), in which each rule's consequent is specified by a MF of a step function center at the constant. Figure 4 illustrate the reasoning mechanism for zero-order Sugeno model. This class of fuzzy models should be used when only performance is the ultimate goal of predictive modeling as in the case of our modeling methodology. This class of fuzzy models can employ all the other types of fuzzy reasoning mechanisms because they represent a special case of each of the above described fuzzy models. More specifically, the consequent part of this simplified fuzzy rule can be seen either as a singleton fuzzy set in the Mamdani model or as a constant output function in TS models. Thus the two fuzzy models are unified under this simplified fuzzy model. Different types of membership functions can be used for the antecedent fuzzy sets. In this work, the membership functions have been tested based on error analysis (calculation of average error). The membership function with minimum error is selected and that will be the suitable membership function to estimate the model. Therefore, triangular-shaped membership functions are used for zero-order TSK based models in this study. Based on a set of K rules, the output for any unknown input vector $x(0)$ is obtained by the following fuzzy reasoning procedure:

- Calculate the degree of fulfillment for the k-*th* rule, for k = 1,…,K, by means of Larsen product operator:

$$\mu_k(X) = \prod_{i=1}^{n} \mu_{ik}(x_i), \qquad k = 1,......,K \tag{7}$$

Note that when computing the activation strength of a rule, the connective AND can be interpreted through different T-norm operators: typically there is a choice between product and min operators. Here we choose the product operator because it retains more input information than the min operator and generally gives a smoother output surface which is a desirable property in any modeling application.

- Calculate the inferred outputs $\hat{y}_j$ by taking the weighted average of consequent values B_j^k with respect to rule activation strengths $\mu_k(x)$:

$$\hat{y}_j = \frac{\sum_{k=1}^{K} \mu_k(X) b_{jk}}{\sum_{k=1}^{K} \mu_k(X)}, \qquad j = 1,....,m \tag{8}$$

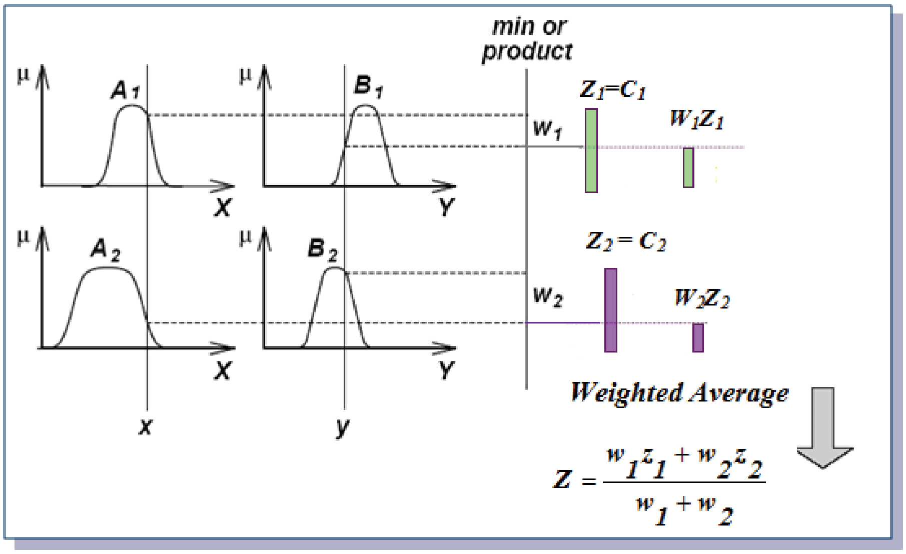

Fig. 4. Zero-order TSK fuzzy inference system with two inputs and two rules (Castillo & Melin, 2001).

6.3.1 Parameter selection for the system

For a real-world modeling problem, it is not uncommon to have tens of potential inputs to the model under construction. An excessive number of inputs not only impair the transparency of the underlying model, but also increase the complexity of computation necessary for building the model. Therefore, it is necessary to do input selection that finds the priority of each candidate inputs and uses them accordingly. Specifically, In order to build a reasonably accurate model for prediction, proper parameters must be selected. The MATLAB function exhsrch performs an exhaustive search within the available inputs to select the set of inputs that most influence the desired output. The first parameter to the function specifies the number of input combinations to be tried during the search. Essentially, exhsrch builds an ANFIS model for each combination and trains it for one epoch and reports the performance achieved. The following are some practical considerations in parameter selection:

- Remove some irrelevant inputs such as the type of dialysate, dialysate temperature, blood pressure of patients, probability of complications, blood volume of patients, intercompartmental urea mass transfer area coefficient, fraction of ultrafiltrate from ICF and access blood flow. This was performed based on the recommendations of an expert in the hemodialysis field. This expert is the medical consultant who supervises the dialysis sessions throughout the research.
- Remove inputs that can be derived from other inputs.
- Make the underlying model more concise and transparent.
- The reduction of the number of parameters results in the reduction of the time required for model construction.
- The selected parameters must affect the target problem, i.e., strong relationships must exist among the parameters and target (or output) variables.
- The selected parameters must be well-populated, and corresponding data must be as clean as possible.

The proposed input selection method is based on the assumption that the ANFIS model with the smallest RMSE (root mean squared error) after one epoch of training has a greater potential of achieving a lower RMSE when given more epochs of training. This assumption is not absolutely true, but it is heuristically reasonable. For instance, if we have a modeling problem with ten candidate inputs and we want to find the most three influential inputs as the inputs to ANFIS, we can construct C_3^{10} =120 ANFIS models, each with different combination of inputs and train them with a single pass of the least-squares method. The ANFIS model with the smallest training error is then selected for further training using the hybrid learning rule to tune the membership functions as well. Note that one-epoch training of 120 ANFIS models in fact involves less computation than 120-epoch training of a single ANFIS model, therefore the input selection procedure is not really as computation intensive as it looks. Therefore, five inputs are selected as the data set for C_{eq} predictor. They are, urea pre-dialysis (Cpre, mg/dl) at the beginning of the procedure, urea post-dialysis (C_{post}, mg/dl), Blood flow rate (BFR, dl/min), desired dialysis Time (T_d, min) and Ultrafiltration rate, the removal of excess water from the patient (UFR, dl/min). All blood samples were obtained from the arterial line at different times for urea determinations. The ANFIS output is the equilibrated post-dialysis BUN (C_{eq}) which was obtained 30 and 60 minutes after HD. Two triangular membership functions (MFs) are assigned to each linguistic variable. The ANFIS structure containing 5^2 = 32 fuzzy rules and 92 nodes. Each fuzzy rule is constructed through several parameters of membership function in layer 2 with a total of 62 fitting parameters, of which 30 are premise (nonlinear) parameters and 32 are consequent (linear) parameters. To achieve good generalization capability, it is important that the number of training data points be several times larger than the number parameters being estimated. In this case, the ratio between data and parameters is five (310/62). Once the FIS structure was identified, the parameters that had to be estimated (Triangular input MF parameters and output constants) were fitted by the hybrid-learning algorithm.

6.4 Training methodology of the developed ANFIS system

The core of the ANFIS calculations was implemented in a MATLAB environment. Functions from the Mathwork's MATLAB Fuzzy Logic Toolbox (FLT) were included in a MATLAB

code programmed by the author[1] to solve the input-output problem with different numbers of input MFs, using all data available. An estimate of the mean square error between observed and modeled values were computed for each trial, and the best structure was determined considering a trade-off between the mean square error and the number of parameters involved in computation.
Input MFs were linked by all possible combinations of if-and-then rules defining an output constant for each rule. The flow chart of proposed training methodology of ANFIS system is shown in Fig. 5. The modeling process starts by obtaining a data set (input-output data pairs) and dividing it into training and checking data sets. Training data constitutes a set of input and output vectors. The data is normalized in order to make it suitable for the training process. This normalized data was utilized as the inputs and outputs to train the ANFIS. To avoid overfitting problems during the estimation, the data set were randomly split into two sets: a training set (70% of the data; 220 samples), and a checking set (30% of the data; 90 samples). When both checking data and training data were presented to ANFIS, the FIS was selected to have parameters associated with the minimum checking data model error. In other words, two vectors are formed in order to train the ANFIS, input vector and the output vector (Fig. 5). The training data set is used to find the initial premise parameters for the membership functions by equally spacing each of the membership functions. A threshold value for the error between the actual and desired output is determined. The consequent parameters are found using the least-squares method. Then an error for each data pair is found. If this error is larger than the threshold value, update the premise parameters using the gradient decent method. The process is terminated when the error becomes less than the threshold value. Then the checking data set is used to compare the model with actual system. A lower threshold value is used if the model does not represent the system. Training of the ANFIS can be stopped by two methods. In the first method, ANFIS will be stopped to learn only when the testing error is less than the tolerance limit. This tolerance limit would be defined at the beginning of the training. It is obvious that the performance of a ANFIS that is trained with lower tolerance is greater than ANFIS that is trained with higher tolerance limit. In this method the learning time will change with the architecture of the ANFIS. The second method to stop the learning is to put constraint on the number of learning iterations.

6.5 Testing and validation process of the developed ANFIS

Once the model structure and parameters have been identified, it is necessary to validate the quality of the resulting model. In principle, the model validation should not only validate the accuracy of the model, but also verify whether the model can be easily interpreted to give a better understanding of the modeled process. It is therefore important to combine data-driven validation, aiming at checking the accuracy and robustness of the model, with more subjective validation, concerning the interpretability of the model. There will usually be a challenge between flexibility and interpretability, the outcome of which will depend on their relative importance for a given application. While, it is evident that numerous cross-validation methods exist, the choice of the suitable cross-validation method to be employed in the ANFIS is based on a trade- off between maximizing method accuracy and stability

[1] The ANFIS source code developed by the author for training the system is copyright protected and not authorized for sharing.

and minimizing the operation time. In this research, the hold-out cross-validation method is adopted for ANFIS because of its accuracy and possible implementation. The choice of the hold-out method is attributed to its relative stability and low computational time requirements. A major challenge in applying the temporal cross-validation approach is the need to select the length of the checking data set utilized. Different lengths of the cross-validation data set ranging from one tenth to one third of the window size were examined. Apparently, choosing one third of the original data lead to short data set for the training process that may cause difficulty to reach the error goal.

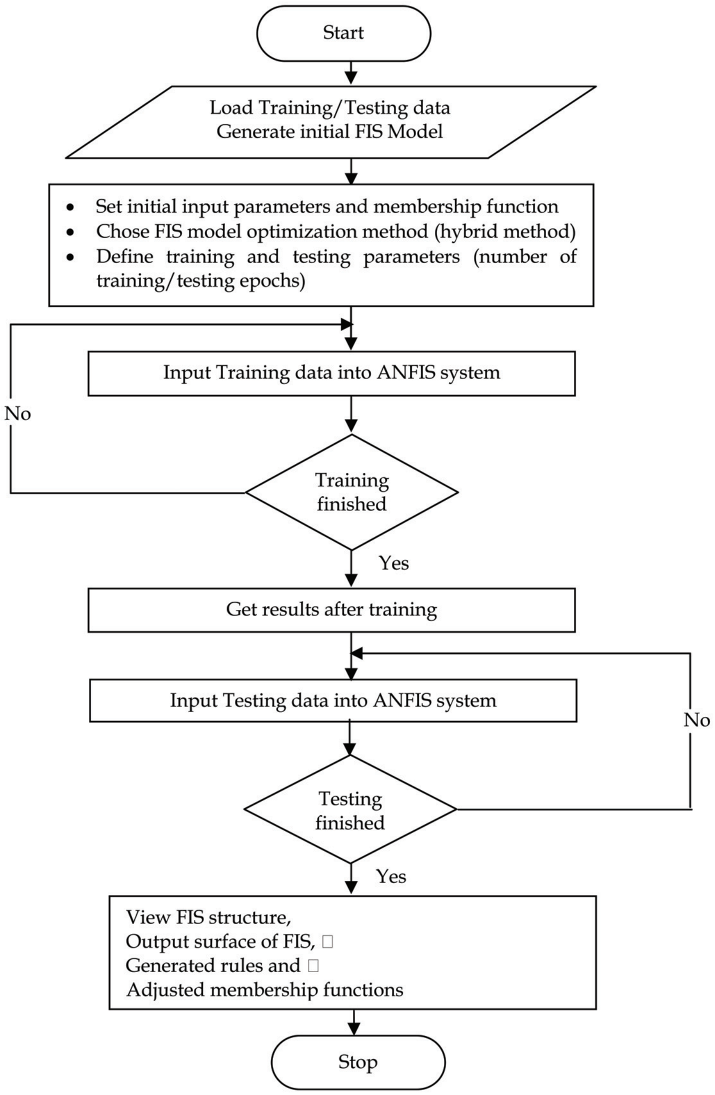
Start
Load Training/Testing data
Generate initial FIS Model
• Set initial input parameters and membership function
• Chose FIS model optimization method (hybrid method)
• Define training and testing parameters (number of training/testing epochs)
Input Training data into ANFIS system
No
Training finished
Yes
Get results after training
Input Testing data into ANFIS system
No
Testing finished
Yes
View FIS structure,
Output surface of FIS,
Generated rules and
Adjusted membership functions
Stop

Fig. 5. Flow chart of training methodology of ANFIS system.

While choosing 20% of window size lead to weakness in detecting the features of the expected data set in prediction stage since it leads to relatively short data set for the cross-validation procedure. Therefore, it was decided to select the length of data set for cross-validation utilized in our study to be 30% of the original data-set. Two pair sets were made with different combinations of 70% and 30% of the samples to improve the generalization properties of the adopted ANFIS as follows:

- Pair set 1: training set first 70%; test set last 30%
- Pair set 2: training set last 70%; test set first 30%

For each pair set, two ANFIS models of the same size, but differing in initialization weights, were trained to study the stability and robustness of the each model. The best weights (giving minimum mean-squared error) of two different training sessions over each input/output training set were chosen as the final ANFIS models. The performances of the ANFIS models both training and testing data are evaluated and the best training/testing data set is selected according to mean absolute error (MAE), mean absolute percentage error (MAPE), Root Mean Square Error (RMSE) and normalized root mean squared error (NRMSE). Prediction accuracy is calculated by comparing the difference of predicted and measured values. If the difference is within tolerance, as in $|C_{eq}$ predicted-C_{eq} measured$| \leq \varepsilon$, accurate prediction is achieved. The tolerance ε is defined based based on the recommendations of an expert in the hemodialysis field. In equilibrated blood urea concentration (C_{eq}) prediction, errors of ±1.5% are allowed. So the prediction accuracy is defined as follows:

$$Accuracy = \frac{\left|C_{eq}^{predicted} - C_{eq}^{measured}\right| \leq \varepsilon}{\|predicted\ set\|} \tag{9}$$

For the five input parameters, each one was assigned two fuzzy sets, i.e. low and high. The membership function μ(k) for each input parameter is divided into two regions, low and high. The number and type of parameters for training ANFIS are shown in table 4.

ANFIS parameter type	**Value**
TSK Type	Zero-order
Numbers of Rules	32
Number of Training Epochs	50
Number of nodes	92
Total fitting parameters	62
• premise (nonlinear) parameters	30
• consequent (linear) parameters	32
Number of Membership functions	Traingular-2
Defuzzification Method	Weighted average
Initial step size, k_{ini}	0.07
Step increasing rate, η	1.6
Step decreasing rate, γ	0.1

Table 4. Training parameters of C_{eq30} ANFIS prediction model.

The collection of well-distributed, sufficient, and accurately measured input data is the basic requirement in order to obtain an accurate model. The data set is divided into separate data sets- the training data set and the test data set. The training data set is used to train the ANFIS, whereas the test data set is used to verify the accuracy and effectiveness of the trained ANFIS. The ANFIS was tuned using a hybrid system that contained a combination of the back propagation and least-squares-type methods. An error tolerance of 0 was used and the ANFIS was trained with 50 epochs. After training and testing, the RMSE became steady, the training and testing were regarded as converged as shown in Fig. 6. RMSE from each of the validating epochs was calculated and averaged to give the mean RMSE. The network error convergence curve achieved mean RMSE values of 0.2978 and 0.3125 for training and testing, respectively. It is noted from error curves that the ANFIS model performed well and it is obvious that the error between the actual and the predicted output of the model is very insignificant.

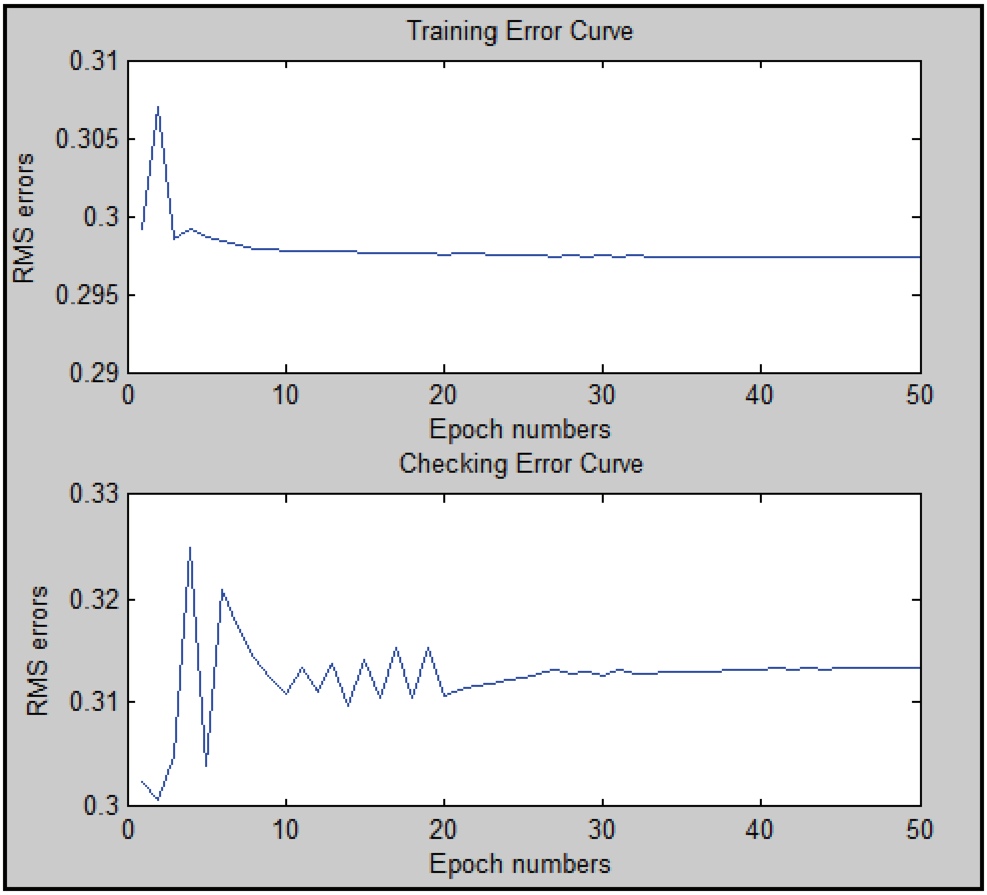

Fig. 6. Training and checking errors obtained by ANFIS for predicting C_{eq30}.

The final member functions of the five inputs are changed through supervised learning. In a real world domain, all the features described may have different levels of relevancy. Moreover, human-determined membership functions are seldom optimal in terms of producing desired outputs. Therefore, the training data are sufficient to provide the available input-output data necessary for fine-tuning the membership function. Figure 7 shows the final membership functions of the input parameters derived by training via the triangular membership function. Considerable changes happened in the final membership function after training especially for ultrafiltration rate (UFR) input. After the training process, the model is validated by comparing the predicted results against the experimental data. The validation tests between the predicted results and the actual results for both training and testing phases are summarized in table 5. The statistical analysis demonstrated that there is no statistically significant difference was found between the predicted and the measured values. The percentage of MAE and RMSE for testing phase is 0.44% and 0.61% respectively.

The same data set were used for predicting equilibrated urea concentration at 60 min (C_{eq60}) post-dialysis session. The C_{eq60} model achieved RMSE values of 0.2707 and 0.3125 for training and testing, respectively. The results obtained indicate that ANFIS is a promising

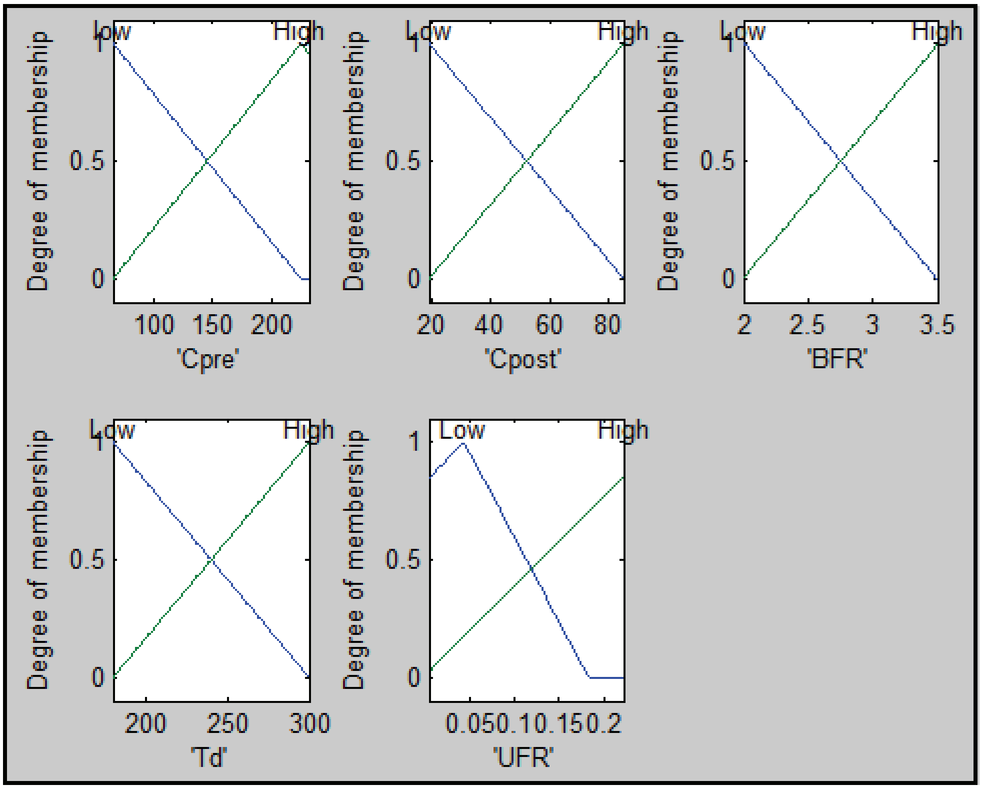

Fig. 7. The final membership functions of selected inputs for C_{eq30} predictor.

Agreement Comparison	$C_{eq30\text{-measured}}$ **versus** $C_{eq30\text{-ANFIS}}$	
	Training Phase	**Testing Phase**
Mean Absolute Error (MAE)	0.2383	0.2425
MAPE	0.0044	0.0044
Root mean square error (RMSE)	0.2978	0.3125
Normalized root mean square Error (NRMSE)	0.0040	0.0061
Median algebraic difference Δ	0.2347	0.0113
Median absolute difference |Δ|	0.6529	0.3143

Table 5. Validation tests between the measured C_{eq30} as a reference and the predicted one by the ANFIS system.

tool for predicting equilibrated urea concentration at 30 min (C_{eq30}) and 60 min (C_{eq60}) post-dialysis session. Both C_{eq30} and C_{eq60} models had very low MAE and RMSE values for both training and testing. This model was conducted to determine how the equilibrated urea concentration (C_{eq}) could be predicted without having to take a final urea sample an hour after the patient had completed the dialysis session.

7. Conclusion

Predictive modeling is the process of identifying a model of an unknown or complex process from numerical data. Due to the inherent complexity of many real processes, conventional modeling techniques have proved to be too restrictive. Recently, the hybrid approach to predictive modeling has become a popular research focus. A novel hybrid system combining different soft computing paradigms such as neural networks and fuzzy systems has been developed for predictive modeling of dialysis variables in order to estimate the equilibrated dialysis dose ($_{eq}$Kt/V), without waiting for 30-60 min post-dialysis to get the equilibrated urea sample which is inconvenient for patients and costly to the dialysis unit. The aim of using a neuro-fuzzy network is to find, through learning from data, a fuzzy model that represents the process underlying the data. In neuro-fuzzy models, connection weights, propagation and activation functions differ from common neural networks. Although there are a lot of different approaches, the term neuro-fuzzy is restricted to systems which display the following properties:

- A neuro-fuzzy system is a fuzzy system that is trained by learning algorithm (usually) derived from neural network theory. The (heuristic learning procedure operates on local information, and causes only local modifications in the underlying fuzzy system. The learning process is not knowledge based, but data driven.
- A neuro-fuzzy system can be viewed as a special 3-layer feedforward neural network. The units in this network use *t*-norms or *t*- cononrms instead of the activation functions usually used in neural networks.

The first layer represents input variables, the middle (hidden) layer represents fuzzy rules and the third layer represents output variables. Fuzzy sets are encoded as (fuzzy) connection weights. Some neuro-fuzzy models use more than 3 layers, and encode fuzzy sets as activation functions. In this case, it is usually possible to transform them into 3-layer architecture. This view of fuzzy systems illustrates the data flow within the system and its parallel nature. However this neural network view is not a prerequisite for applying a learning procedure, it is merely a convenience.

- A neuro-fuzzy system can always (i.e. before, during and after learning) be interpreted as a system of fuzzy rules. It is both possible to create the system out of training data from scratch, and it is possible to initialize it by prior knowledge in form of fuzzy rules.
- The learning procedure of a neuro-fuzzy system takes the semantical properties of the underlying fuzzy system into account. This result in constraints on the possible modifications of the system's parameters.
- A neuro-fuzzy system approximates an n-dimensional (unknown) function that is partially given by the training data. The fuzzy rules encoded within the system represent vague samples, and can be viewed as vague prototypes of the training data. A neuro-fuzzy system should not be seen as a kind of (fuzzy) expert system, and it has nothing to do with fuzzy logic in the narrow sense.

Besides accuracy, model transparency is the most important goal of the proposed modeling methodology, which supports this feature at different levels. The first level of transparency supported is the ability to represent the knowledge characterizing the relations between the data in a natural manner, e.g. as a series of linguistic fuzzy rules, which is common to all neuro-fuzzy approaches. The second level of transparency required to models derived from data is a simple structure. Indeed, the use/reliability of a fuzzy model can become limited with too many fuzzy rules. Hence a compromise between model complexity and model accuracy should be found by identifying parsimonious structures, which aid handling and comprehension of the fuzzy rule base. Simple structures also could be beneficial in improving the model generalization capabilities. A further level of model transparency is a truly linguistic representation of the produced fuzzy rules. By supporting these various levels of transparency, the proposed neuro-fuzzy modeling methodology significantly aids the process of knowledge discovery and model validation. With a data-driven methodology, like the proposed one, fundamental to the success of the modeling process is the availability of good empirical data. When data is limited and/or poorly distributed, the modeling task can easily become unmanageable. This reinforces the importance of the human for injecting a priori knowledge, expert judgment and intuition in to the modeling process. The developed methodology enables the incorporation of a-priori knowledge into the modeling process so as to compensate for the lack of data. When a-priori knowledge provided by the expert takes the form of qualitative descriptions of the process underlying the data, it can be easily inserted into the modeling process by building and pre-weighting the neuro-fuzzy network, giving an initial model which can be later refined in the presence of training data. These are the more appealing features of the proposed methodology.

8. References

Azar, A.T.; Kandil, A.H.; Wahba, K. and Massoud, W. (2008a). Prediction Of Post-dialysis Blood Urea Concentration Using Artificial Neural Network. *Proceedings of 2nd*

International conference on Advanced Control Circuits & systems (ACCS'08), March 30-April 2, 2008, Cairo, Egypt.

Azar, A.T.; Kandil, A.H.; Wahba, K. and Massoud, W. (2008b). Neuro-Fuzzy System for Post-dialysis Urea Rebound Prediction, *Proceedings of IEEE 4th Cairo International Biomedical Engineering Conference (CIBEC'08)*, Dec. 18-20, 2008, Cairo, Egypt.

Azar, A.T.; Kandil, A.H.; Wahba, K. (2009a). Artificial Intelligence for Prediction of Equilibrated Dialysis Dose without Intradialytic Sample. *Saudi J Kidney Dis Transplant*, to be published.

Azar, A.T. (2009b). *Adaptive Neuro-Fuzzy System for Hemodialysis Treatment Process*, Ph.D. dissertation, Dept. Sys. & Biomed. Eng., Cairo Univ., Egypt.

Berenji, R.H. (1992). A reinforcement learning-based architecture for fuzzy logic control. *International Journal of Approximate Reasoning*, Vol. 6, Issue 2.

Bersini H.; Nordvik, J.P & Bonarini, A. (1993). A simple direct adaptive fuzzy controller derived from its neutral equivalent, *Proceedings of 2nd IEEE International Conference on Fuzzy Systems*, Vol. 1, pp. 345-350.

Bhaskaran, S.; Tobe, S.; Saiphoo, C.; Moldoveanu, A.; Raj, D.S. and Manuel, M.A. (1997). Blood urea levels 30 minutes before the end of dialysis are equivalent to equilibrated blood urea, *ASAIO J*, Vol. 43, No. 5, pp. M759-762.

Brown M. & Harris C.J. (1995). *Neurofuzzy Adaptive Modelling and Control*. Prentice Hall, 1st edition, Hemel Hempstead.

Buckley, J.J. & Eslami, E. (1996). Fuzzy Neural Networks: Capabilities. In *Fuzzy Modeling: Paradigms and Practice*, Pedrycz W, ed., pp. 167-183, Kluwer, Boston,

Canaud, B.; Bosc, J.Y; Leblanc, M.; Garred, L.; Vaussenat, F.; Bonardet, A. and Mion, C. (1997). A simple and accurate method to determine equilibrated post-dialysis urea concentration, *Kidney Int*, Vol. 51, No. 6, pp. 2000-2005.

Castillo, O & Melin, P. (2001). *Soft Computing for Control of Non-Linear Dynamical Systems*. 1st edition, Physica-Verlag Heidelberg, Germany.

Cho, K. & Wang, B. (1996). Radial basis function based adaptive fuzzy systems and their application to system identification and prediction. *Fuzzy sets and systems*, Vol. 83, No. 3, pp. 325–339.

Cox, E. (1994). *The Fuzzy Systems Handbook*. AP Professional - New York.

Daugirdas, J.T. (1993). Second generation logarithmic estimates of single-pool variable volume Kt/V: An analysis of error. *JAm Soc Nephrol*, Vol. 4, pp. 1205-1213.

Daugirdas, J.T.; Burke, M.S.; Balter, P.; Priester-Coary, A. and Majka, T. (1996). Screening for extreme postdialysis urea rebound using the Smye method: patients with access recirculation identified when a slow flow method is not used to draw the postdialysis blood. *Am J Kidney Dis*, Vol. 28, No. 5, pp. 727-731.

Daugirdas, J.T & Depner, T.A. (1994). A nomogram approach to hemodialysis urea modeling. *Am J Kidney Dis*, Vol. 40, pp. 23-33.

Daugirdas, J.T & Schneditz, D. (1995). Overestimation of hemodialysis dose depends on dialysis efficiency by regional blood flow but not by conventional two pool urea kinetic analysis. *ASAIO J*, Vol. 41, pp. M719-M724.

Depner, T.A. (1994). Assessing Adequacy Of Hemodialysis Urea Modeling. *Kidney Int*, Vol., 45, No. 5, pp. 1522-1535.

Depner, T.A. (1999). History of Dialysis Quantitation, *Semin Dial*, Vol. 12, No.1, pp. S14- S19.

Fuller, R. (2000). *Introduction to Neuro-Fuzzy Systems*. Advances in Soft Computing Series, Springer-Verlag, Berlin/Heildelberg.

Gotch, F.A. & Sargent, J.A. (1985). A Mechanistic Analysis of the National Cooperative Dialysis Study, *Kidney Int*, Vol. 28, No. 3, pp. 526-538.

Guh, J.; Yang, C.; Yang, J.; Chen, L. and Lai, Y. (1998). Prediction of equilibrated postdialysis BUN by an artificial neural network in high-efficiency hemodialysis. *Am J Kidney Dis*, Vol. 31, pp. 638- 646.

Haykin, S. (1998). *Neural Networks, A Comprehensive Foudation*. Second Edition, Prentice Hall.

Ichihashi, H. and Turksen, I. (1993). A neuro-fuzzy approach to data analysis of pairwise comparisons. Int. *Journal of Approximate Reasoning*, Vol. 9, No.3, pp. 227–248.

Jang, J.S.R. & Sun, C.T. (1993). Functional Equivalence Between Radial Basis Function Networks and Fuzzy Inference Systems, *IEEE Trans. on Neural Networks*, Vol. 4, No. 1, pp. 156-159.

Jang, J.S.R. (1993). ANFIS: adaptive-network-based fuzzy inference system, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 23, No.3, pp. 665–685.

Jang, J.S.R. & Sun, C.T. (1995). Neuro-Fuzzy Modeling and Control, *Proceedings of the IEEE*, Vol. 83, pp. 378-406.

Jang, J.S.R; Sun, C.T & Mizutani, E. (1997). *Neuro-Fuzzy and soft computin*. Prentice-Hall: Englewood Cliffs, NJ.

Jin, Y.; Von Seelen, W.; and Sendhoff, B. (1999). On Generating FC Fuzzy Rule Systems from Data Using Evolution Strategies. *IEEE Trans. on Systems, Man and Cybernetics - Part B: Cybernetics*, Vol. 29, No. 6, pp. 829-845.

Lin, C.T. & Lee, C.S. (1991). Neural-Network-Based Fuzzy Logic Control and Decision Systems. *IEEE Trans. on Computers*, Vol. 40, No. 12, pp. 1320-1336.

Lin, C.T. and Lee, G. (1996). *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Ed. Prentice Hall.

Maduell, F.; Garcia-Valdecasas, J.; Garcia, H.; Hernandez-Jaras, J.; Sigüenza, F.; del Pozo, C.; Giner, R.; Moll, R.; and Garrigos, E. (1997). Validation of different methods to calculate Kt/V considering post-dialysis rebound. *Nephrol Dial Transplant*, Vol. 12, No. 9, pp. 1928-1933.

Mamdani, E.H. & Assilian, S. (1975). An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller, *International Journal of Man-Machine Studies*, Vol. 7, No. 1, pp. 1-13.

Mehrotra, K., Mohan, C. K., and Ranka, S. (1997). *Elements of Artificial Neural Networks*. The MIT Press.

Nauck, D. (1994). A fuzzy perceptron as a generci model for neuro-fuzzy approaches. *Proceedings of Fuzzy-Systeme'94*, 2nd GI-Workshop.

Nauck, D & Kruse, R. (1998a). A Neuro-Fuzzy Approach to Obtain Interpretable Fuzzy Systems for Function Approximation. *Proceedings of the IEEE International Conference on Fuzzy Systems*, Anchorage, AK, pp. 1106-1111.

Nauck, D & Kruse, R. (1998b). How the Learning of Rule Weights Affects the Interpretability of Fuzzy Systems, *Proceedings of the IEEE International Conference on Fuzzy Systems*, Anchorage, AK, Vol.2, pp. 1235- 1240.

Nauck, D. & Kruse, R. (1999). Neuro-fuzzy systems for function approximation. *Fuzzy Sets and Syst.*, Vol. 101, pp. 261-271.

Nomura, H., Hayashi, I., and Wakami, N. (1992). A self-tuning method of fuzzy control by descent method. *Proceedings of IEEE International Conferenceon Fuzzy Systems*, pp. 203–210.

Royas, I.; Pomares, H.; Ortega, J.; and Prieto, A. (2000). Self-Organized Fuzzy System Generation from Training Examples, *IEEE Trans. on Fuzzy Systems*, Vol. 8, No. 1, pp. 23-36.

Ruspini, E., Bonissone, P., and Pedrycz, W. (1998). *Handbook of Fuzzy Computation*. Ed. Iop Pub/Inst of Physics.

Schneditz, D. & Daugirdas, J.T. (2001). Compartment effects in hemodialysis, *Semin Dial*, Vol. 14, No. 4, pp. 271-277.

Seng, T.L.; Khalib, M.B; and Yusof, R. (1999). Tuning of a Neuro-Fuzzy Controller by Genetic Algorithm. *IEEE Trans. on Systems, Man and Cybernetics-Part B, Cybernetics*, Vol. 29, No. 2.

Shi, Y. and Mizumoto, M. (2000a). A new approach of neurofuzzy learning algorithm for tuning fuzzy rules. *Fuzzy sets and systems*, 112(1):99–116.

Shi, Y. and Mizumoto, M. (2000b). Some considerations on conventional neuro-fuzzy learning algorithms by gradient descent method. *Fuzzy sets and systems*, Vol. 112, No. 1, pp. 51–63.

Smye, S.W.; Dunderdale, E.; Brownridge, G. and Will, E. (1994). Estimation of treatment dose in high-efficiency hemodialysis, *Nephron*, Vol. 67, No. 1, pp. 24-29.

Takagi, H. & Hayashi, I. (1991). NN-driven fuzzy reasoning. *International Journal of Approximate Reasoning*, Vol. 5, Issue 3.

Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans Syst Man Cybern*, Vol. 15, No. 1, pp. 116-132.

Tattersall, J.E; DeTakats, D., Chamney, P., Greenwood, RN. and Farrington, K. (1996). The post-hemodialysis rebound: Predicting and quantifying its effect on Kt/V. *Kidney Int*, Vol. 50, pp. 2094-2102.

Tsukamoto, Y. (1979). An approach to fuzzy reasoning method. In *Advances in fuzzy set theory and applications,* M.M. Gupta, R.K. Ragade and Yager R.R., Editors , Elsevier, North-Holland.

Wang, L. and Mendel, J. (1992). Back-propagation fuzzy system as nonlinear dynamic system identifiers. *Proceedings of IEEE International Conferenceon Fuzzy Systems*, pages 1409–1416.

Yager, R. and Filev, D. (1994). Generation of fuzzy rules by mountain clustering. *Journal of Intelligent Fuzzy Systems*, Vol. 2, No. 3, pp. 209–219.

Yashiro, M., Watanabe, H. and Muso, E. (2004). Simulation of post-dialysis urea rebound using regional flow model. *Clin Exp Nephrol*, Vol. 8, No. 2, pp. 139-45.

Yen, J. & Wang, L. (1999). Simplifying Fuzzy Rule-Based Models Using Orthogonal Transformation Methods. *IEEE Trans. on Systems, Man and Cybernetics-Part B: Cybernetics*, Vol. 29, No. 1, pp. 13-24.

Zadeh, L. (1965). Fuzzy sets. *Inf Cont*, Vol. 8, pp. 338–353.