Fuzzy inferencing using single-antecedent fuzzy rules

Sebastian W. Khor, M. Shamim Khan, and Kok Wai Wong

Abstract

The output of a fuzzy cognitive map (FCM) is the summation of the products of its individual input variables and their relative weights to each node. This additive nature is the main hindrance to the implementation of fuzzy theory in FCM, as conventional fuzzy rules rely on the mapping of the input spaces to the output space via the intersection configuration of fuzzy rules. On the other hand, fuzzy rules are well known for the combinatorial rule explosion problem. We present a methodology that allows the use of single-antecedent fuzzy rules instead of multiple-antecedent fuzzy rules. This methodology thus can be implemented in FCM and at the same time eliminate altogether the problem associated with the intersection of fuzzy rules. Our proposed method ensures transparency of the rules, unlike in conventional data-driven fuzzy systems where they can become ambiguous.

Keywords: Fuzzy Knowledge Map, Fuzzy logic, fuzzy modelling, single-antecedent fuzzy rules.

1. Introduction

Introduced by Robert Axelrod, cognitive map [1] was designed to model and study the causal beliefs of decision makers, and to improve the quality of decision-making processes, using positive and negative symbols to represent positive and negative causality respectively. Later B. Kosko extended the Cognitive Map to Fuzzy Cognitive Map (FCM) to simulate causal relationships between concepts [2, 3] where the degree of causality is expressed in the interval [0, 1] with 1 representing full strength of causality, 0 indicating absence of any causal effect. Since then, various attempts have been made to extend the FCM so that the causality can be expressed as real or fuzzy values, to model a number of application areas such as policy analysis [4], and social systems [5]. Hagiwara [6] proposed non-linear and time-delay arcs; Satur and Liu[7] introduced the contextual FCM for geographic information systems. However, despite these developments, there still exist several limitations. FCMs can only express casual relationships, thus limiting their scope of applications.

Carvalho and Tomé [8] proposed a rule based fuzzy cognitive map (RBFCM) where fuzzy rules are implemented for reasoning. However, these rules conform weakly to the original fuzzy theory. Like FCM, RBFCM cannot solve problems associated with multi-antecedent fuzzy rules.

In this paper we show that problems that can be expressed as polynomial or continuous functions can be approximated by single-antecedent fuzzy rules. We also show that our proposed methodology can be used to implement fuzzy rules in FCM and express both causal as well as non-causal relations, thus overcoming the limitations in FCM and RBFCM.

We discuss the limitations of FCM in Section II and RBFCM and the rule problem in Section III. We present our proposed methodology to overcome these problems in Section IV and its construction in Section V. In Section VI we present a simulation example and experiment, followed by our conclusion in Section VII.

2. Fuzzy cognitive map

A Fuzzy Cognitive Map or FCM (Figure 1) is a directed graph used to model a collection of concepts or events (usually depicted as circles or nodes) and the relations (depicted as directed arcs or edges) between these nodes. Nodes represent system variables; their values change over time. Arcs represent causality, the direction of which indicates the direction of causality or reasoning. As shown in Figure 1, an arc connects N_1 (also called the antecedent node) to N_2 (the consequent node), indicating that N_1 causes change in N_2. Arcs take on values or weights, representing degrees of causality, usually in the interval [-1, 1]. When the weight is 0, the antecedent node has no influence on the consequent node, in which case, the arc between the nodes is not shown in the graph

When the causality is non-linear, a transformation function is used. The common functions used are the step, sigmoid, and the tanh functions.

In Figure 1, the path N_4—N_2—N_3—N_4 constitute a

Corresponding Author: Khor S. is with the Department of ARTS, School of Information Technology, Murdoch University, GPO Box S1400, Perth WA 6849 Australia.
E-mail: s.khor@ieee.org
Manuscript received 10 Mar 2006; accepted 20 June. 2006.

feedback loop. The feedback mechanism allows dynamic expression of knowledge over time sequences. Where feedback is absent, the FCM is said to be trivial [9], in which case the FCM inferencing is static (i.e. not cyclical) and simulation will terminate after the first iteration. Thus, FCMs are often regarded as a paradigm similar to that of artificial neural networks (ANN). Typically, neural networks contain many arcs (usually in the 10s and some cases 100s) joining the nodes together. The assumption is that, with a large number of nodes and their interconnections, the network becomes more tolerant of any distortion in the network. Any distortion in one part of the network is compensated for in other parts. However, FCMs do not have this advantage in that the number of arcs to a node is small, often just one or two. Carvalho [10] observes that feedback loops (which are often present in an FCM, though not mandatory) actually aggravate the errors present in the network as the simulation progresses.

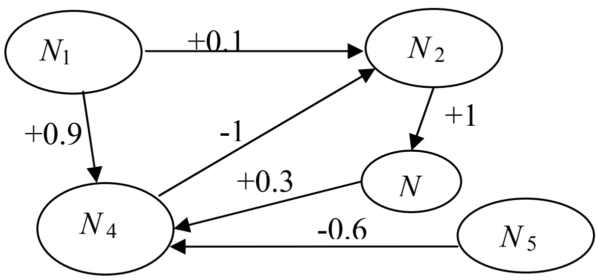

Figure 1. A simple example of a fuzzy cognitive map with feedback.

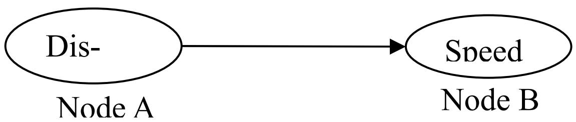

Figure 2. A non-monotonic causal relationship.

Another concern is that FCMs are used to describe the causal relations in a forward-inferencing and monotonic way [11]. For example, if there is a positive causal arc of a certain strength between a antecedent node *A* and an consequent node *B*, the state value of *B* will increase (decrease) with any increase (decrease) in the state value of *A*.

However, in the real world, causal relationships are often non-monotonic. Consider for example the relationship between distance run by an athlete and her running speed as depicted in Figure 2. Initially, the speed will increase with increasing distance, but as more distance is covered, the speed will drop with increasing distance. It is difficult for the traditional FCM to simulate such a situation with just the two concepts and a causal arc.

3. Rule based fuzzy cognitive map

An RBFCM [12] is an FCM where fuzzy rule bases are added between an arc and a consequent node, replacing the weights and non-linear transfer functions.

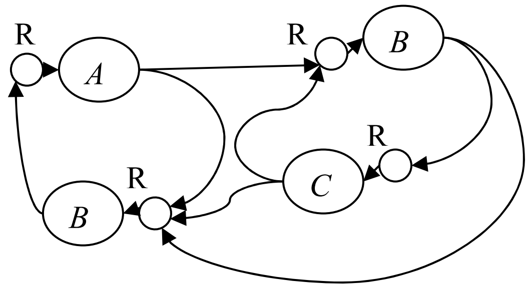

Figure 3. An example of RBFCM. RBs are the rule bases; *A, B, C* and *D* are the nodes.

An example of an RBFCM is shown in Figure 3.

The fuzzy rule in each rule base contains a single membership function. These rules are used to map the input (antecedent node) states to the output (consequent node) states. An RBFCM also has a so-called *fuzzy carry accumulation* mechanism. At each time cycle the fuzzy inference outputs are accumulated at each node. If the accumulation exceeds certain maximum value, it is 'carried over'. This may apparently happen in a multiple-input-single-output situation where the summation of the outputs to a consequent node may exceed the threshold value. We have some reservation on this treatment and will discuss further in Section 4.

Furthermore, under certain condition, e.g. where a consequent node is linked from two antecedent nodes with different fuzzy membership functions, such as *Increase* and *Increase_Much*, RBFCM does not allow singleton inputs [12]. This greatly constrains the usefulness of the system. In fact, since the fuzzy rule is based on a single fuzzy membership function, RBFCM can only reason monotonically as in FCM.

Thus we conclude that RBFCM conforms only weakly to fuzzy logic at most.

4. Single-antecedent fuzzy rules in FKM

We observe that the main limitation of having a map that implements the original fuzzy theory is due to the difficulty in resolving the problem of implementing multi-antecedent fuzzy rules in the arcs of an FCM. Take a hypothetical example: Suppose we have an FCM as in Figure 4.

A set of traditional fuzzy rules may be as follows:

R_1: If a is A_1 and b is B_1 then y is C_1
R_2: If a is A_2 and b is B_2 then y is C_2
R_3: If a is A_3 and b is B_3 then y is C_3

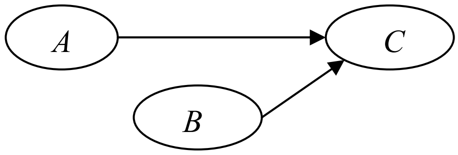

Figure 4. An FCM with two antecedents A and B, and a consequent C.

With such a set of fuzzy rules, spreading the rules over the two arcs is problematic. For example, we may have to decide where the rules should reside – in arc A--C or arc B--C, or somewhere outside the FCM. As more nodes are added, one can image the complexity of the relationships will become. During simulation, we may have to search through all the rules in order to determine which rules get fired. This is exactly the problem of the conventional fuzzy systems we are trying to avoid.

RBFCM is an attempt to solve this problem by providing only a single membership function in each arc. However there are many problems needed to be ironed out. Reference [12] tried to solve these shortcomings by implementing *fuzzy carry accumulation* (FCA) and *shift* operation in the computation of the outputs. For example in our example as in Figure 4, we may have two fuzzy rules:

- If A decreases then C increases;
- If B increases then C increases

Suppose the outputs of the two rules are *Increase* (μ=0.7) and *Increase* (μ=0.5) respectively. Since the sum of these outputs exceeds 1, in FCA it is treated as if an "overflow" has occurred. The overflow value is then "shifted" and carried [8]. Such operations are problematic as agreed by the authors.

We propose using a set of single-antecedent fuzzy rules for each arc instead of a single antecedent rule as in RBFCM. This is implemented in our Fuzzy Knowledge Map (FKM) [13, 14]. It is a directed graph similar to FCM, except that each arc is comprised of a set of single-antecedent fuzzy rules. The direction of an arc indicates the direction of influence or inference. The rules can be derived from experts' knowledge of the problem domain or from data. We will discuss this further in the next section.

Fuzzy systems involve the transformation (or mapping) of inputs to output, using linguistic fuzzy rules of the form:

If x_1 is A_1, and x_2 is A_2, and … and x_k is A_k then y *is* B (1)

where x_i is the ith input to the fuzzy system, which is defined on the universe of discourse X_i; A_i is a fuzzy set on X_i; y is the system output defined on a universe of discourse Y, and B is a fuzzy set on Y. In this paper, we refer to such conjunctive form of the rules as multi-antecedent fuzzy rules, whereas for fuzzy rules of the form:

If x_i is A_i then y is B (2)

we refer to them as single-antecedent fuzzy rules.

The consequent of the rule can also be expressed as a function as in (3)

If x_i is A_i then $y = f(x_i)$. (3)

There are many papers in the literature, which show that Mamdani and Takagi–Sugeno (TS) fuzzy systems are good universal approximators. They can uniformly approximate any continuous functions to any degree of accuracy, e.g.[15-18]. However, evaluating a continuous function using a computer is difficult because the continuous function is computationally expensive and at times may not be readily available or derived at.

The Weierstrass approximation theorem has been widely studied. It states that any continuous function defined on an interval [a, b] can be uniformly approximated as closely as desired by a polynomial function [19].

Given a set of n data points, we can plot a function to pass through these data points in the form of an n - 1 degree polynomial in x, $P_n(x) = \sum_{i=1}^{n} a_i x^i$. Thus, instead of a continuous function we can first derive a polynomial function that approximates the continuous function to a degree of closeness as desired. We can then approximate the polynomial instead. In this paper we will focus our discussion on using single-antecedent fuzzy rules to approximate polynomials and assume that the approach applies equally well to approximating continuous functions since we can derive a polynomial that approximates as close to a continuous function as we desire.

Consider the case where there exist n + 1 data points (x_i, y_i) where no two x_i are the same. Then there exists at most an n degree polynomial of the form $y = P_n(x) = \sum_{i=0}^{n} a_i x^i$.

We can then substitute all the data points in $P_n(x)$ and solve the n + 1 linear simultaneous equations to obtain the coefficients a_i. An alternative method is to make use of Lagrange interpolation to obtain the coefficients. In that case, we can express the polynomial as:

$$\boldsymbol{P_n(x) = L_1(x)y_0 + L_2(x)y_1} + \ldots + \boldsymbol{L_n(x)y_n} = \sum_{i=0}^{n} L_i(x)\, y_i \qquad (4)$$

where

$$L_k(x) = \frac{(x - x_0)(x - x_1)\ldots(x - x_n)}{(x_k - x_0)(x_k - x_1)\ldots(x_k - x_n)} = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{x - x_i}{x_k - x_i}$$

where i, k = 1, 2, . . ., n, and such that

$$L_k(x_i) = \begin{cases} 1 & if \quad k = i \\ 0 & if \quad k \neq i \end{cases}$$

For the case of degree = 1 (i.e. first order or linear) with the polynomial passing through two points (x_1, y_1) and (x_2, y_2), the Lagrange polynomial is given by:

$$\begin{aligned} P_1(x) &= L_1(x)y_1 + L_2(x)y_2 \\ &= \frac{(x - x_2)}{(x_1 - x_2)}y_1 + \frac{(x - x_1)}{(x_2 - x_1)}y_2 \end{aligned} \tag{5}$$

For $x = x_1$, $L_1 = 1$ and $L_2 = 0$, so $P_1(x_1) = y_1$, i.e. it contains (x_1, y_1).
For $x = x_2$, $L_1 = 0$ and $L_2 = 1$, so $P_1(x_2) = y_2$, i.e. it contains (x_2, y_2).

To simplify our discussion, let us define certain class of polynomial as follows:

Definition: A simple multivariate polynomial is defined as a polynomial function that contains more than one variable but no summands that are formed by the product of two or more of these variables. In other words, it contains only monomials, i.e. the individual summands with the coefficients only.

A simple multivariate polynomial can thus be considered as a composite polynomial comprising two or more univariate polynomials.

As there is no summand that is the product of the variables in a simple multivariate polynomial, these variables are said to be *independent* of each other, in the sense that the net contribution of an input is independent of the value of the input of other variables.

A. Case of univariate polynomial

Suppose there exists a set of fuzzy sets that partitions the input space completely, and the membership functions are ordered, consistent, and convex. Then, as can be seen in Figure 5, an input singleton x belongs to one or at most two membership functions. Suppose the membership functions are normal. Then x will have a degree of membership in [0, 1].

Suppose there exists a set of fuzzy sets that partitions the input space completely, and the membership functions are ordered, consistent, and convex. Then, as can be seen in Figure 5, an input singleton x belongs to one or at most two membership functions. Suppose the membership functions are normal. Then x will have a degree of membership in [0, 1].

We can then create a set of single-antecedent fuzzy rules as in (3). Since the fuzzy membership functions are ordered, consistent and convex, referring to Figure 5, an input will trigger the rules in two situations:

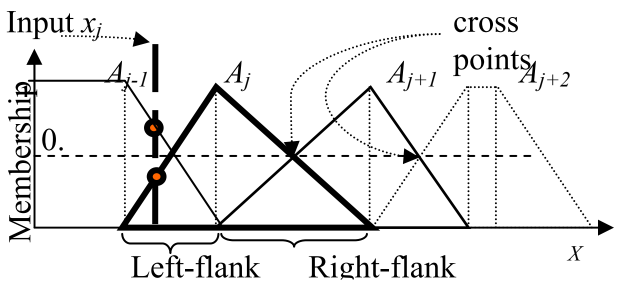

Figure 5. Input singleton x_j cuts membership functions A_{j+1} and A_j at most two points.

1) In the case where the input x cuts at one point in the set of membership functions, the output is determined by the consequent of a single-antecedent fuzzy rule.
   Rule 1: If x is A then y = $f(x)$
2) In the case where the input x cuts at two points, the output is determined by two rules:
   Rule 1: If x is A_1, then y1 = $f_1(x)$
   Rule 2: If x is A_2, then y2 = $f_2(x)$

According to Takagi-Sugeno (TS) fuzzy system, the output is given by:

$$y^*(x) = \frac{w_1 y_1 + w_2 y_2}{w_1 + w_2}. \tag{6}$$

When $\sum_{j=1}^{2} w_j = 1$ for normalized w, (6) becomes:

$$y^*(x) = w_1 y_1 + w_2 y_2. \tag{7}$$

The functions y_1 and y_2 can be constants or linear functions. As would have been noted, this is a case of SISO (single input single output) model. We may set w_1 and w_2 in proportion to the values of x_1 and x_2. For example, $w_1 = (x - x_2)/(x_1 - x_2)$, and $w_2 = (x - x_1)/(x_2 - x_1)$. Substitute these in (7), we have:

$$y^*(x) = \frac{(x - x_2)}{(x_1 - x_2)}y_1 + \frac{(x - x1)}{(x2 - x1)}y_2. \tag{8}$$

The result is the same as (5). That is, the crisp output of a fuzzy system of a univariate input is a point on the polynomial function. In FKM this is represented by a single antecedent node linking to a consequent node by an arc. The output becomes the new state of the consequent node.

B. Case of simple multivariate polynomial

Consider the case of a simple multivariate polynomial with two or more input variables. We know that we can always decompose the multivariate polynomial into univariate polynomial such that each of these polynomials contains only an input variable. We can then form a set of fuzzy rules for each of these univariate polynomials. The sum of the outputs of the sets of these fuzzy rules for each set of inputs yield a point on the original simple multivariate polynomial.

Often we want to express the total output in terms of a normalized output within an interval say, [0, 1]. We may weight the contribution of each term such that the total contributions fall within [0, 1] as given by the equation:

$$y = \beta_1 y^*_1 + \beta_2 y^*_2 \ldots + \beta_n y^*_n \qquad (9)$$

where $\sum_{j=1}^{n} \beta_j = 1$.

β is the weighting of the contribution of each term. In many practical situations, the contributions can be equal, or some terms may have a higher contribution at the expense of others.

In FKM, this is the case where two or more antecedent nodes are linked to a consequent node. The outputs d_i from all the antecedent nodes are aggregated to form the new state of the consequent node.

C. Case of a multivariate polynomial

In a multivariate polynomial, there may be summands that are products of two or more variables. In such cases, it may be difficult to decompose the polynomial into simple multivariate or univariate polynomials. However, in a close interval [a, b], each variable may contribute different values towards the polynomial at different points in the universes of discourse. For example, in the equation: $y = 5x^5 - 9x^4 + x^3z + xz^3$, $5x^5$ dominates the contributions towards output y for large absolute values of x. For variables closer to the origin, the polynomial may wiggle up and down, hitting many highs and lows. Thus, the interval [a, b] determines the overall dominance of the variables. In the example, if [a, b] includes points where x is large, then x dominates. Otherwise either x or z may dominate, or both may equally dominate.

Suppose the contribution by one variable dominates. We can then approximate its contribution, and subtract this from the total contribution to arrive at the net contributions for the rest of the variables. By repeatedly eliminating the contribution of the next dominant input variable, we can eventually evaluate the last variable. We present such a method later in our experiments – a method adopted from [20]. In this paper we will call it the SY method.

Suppose the contributions of two or more input variables are equal, it is still possible to approximate their contributions. We can arbitrarily select a variable, say x, to be the dominant variable and approximate its contribution. Since we arbitrarily choose x to be the dominant variable, excess contribution would have been allotted to x at the expense of other variables. This means that the rest of the variables will also have been allotted less or even negative contributions. The result is that the contributions for all the variables will cancel each other out such that the final output of the model approximates closely to the polynomial function. We will demonstrate this approach in our experiments in Section 6.

It will be worthy to investigate further on other means of decomposing a multivariate polynomial within each partition where the polynomial contains summands of products of terms. For example, it is possible to split the input spaces such that each data point is attributed with a degree of membership that it belongs to certain input space, as in [21, 22]. However, that is out of the scope of this paper. Our experiments however, show that our proposed method produced better results than those of [20] where multiple-antecedent fuzzy rules were used.

5. Construction of fuzzy rules

As in fuzzy systems, we partition the input spaces such that the membership functions are ordered, consistent, and convex, and they share common flanks. Each partition thus forms an interval $[a'_i, b'_i]$ where i is the ith partition. We can then approximate the membership function for each partition. Kosko [3] terms such fuzzy inferencing outputs that are based on partition as fuzzy patches. Figure 6 shows an example of fuzzy patches of a function y with a single variable x.

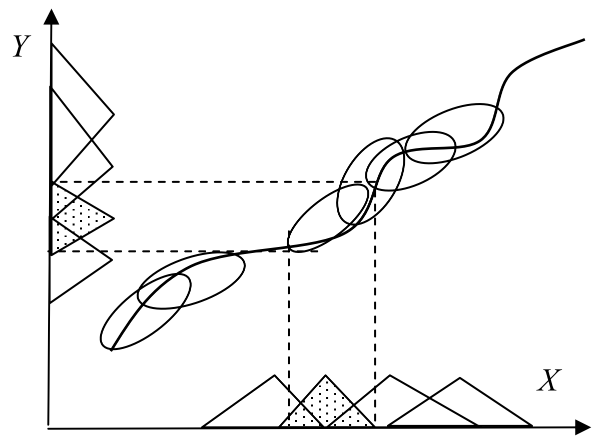

Figure 6. Fuzzy relations in the product space $X \times Y$.

For each membership function, we construct fuzzy rules. In general, fuzzy rules can be constructed in two ways: based on data; and based on expert knowledge.

A. Extraction of fuzzy rules from data

Nowadays data are becoming more and more readily available. Extraction of rules from the data is an important source of knowledge acquisition. We discuss fuzzy rules extraction for the case of single input - single output (SISO) system, and multiple inputs - single output (MISO) system. The case of multiple inputs - multiple outputs (MIMO) systems can be considered as multiple MISO's, and is therefore not discussed further in this paper.

1) Single input single output (SISO)

SISO topology in an FKM is one where there is only a single antecedent node linked to a consequent node, as shown in Figure 7, where N is the antecedent node and M is the consequent node.

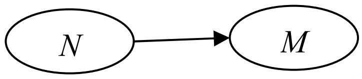

Figure 7. An FKM model of a single input single output (SISO).

The FKM clearly is the case of univariate polynomial. Any input x in Node N is transformed to the output y in Node M by the single-antecedent fuzzy rule as represented by the directed arc linking the two nodes. As there is no difference between the fuzzy rules used in the FKM and the traditional fuzzy rules for a single input variable, any suitable rule extraction methodology can be used. For example, we may use fuzzy c-means clustering to identify the optimal clusters for the input output spaces. For each of these clusters, we construct a fuzzy membership function, upon which we then derive the fuzzy rules.

2) Multiple inputs single output (MISO)

A function of the form of a multivariate polynomial dictates a MISO topology. An example MISO FKM is shown in Figure. 8.

To extract fuzzy rules from a data set containing multiple input variables, we adopted the following steps as proposed in the SY method:

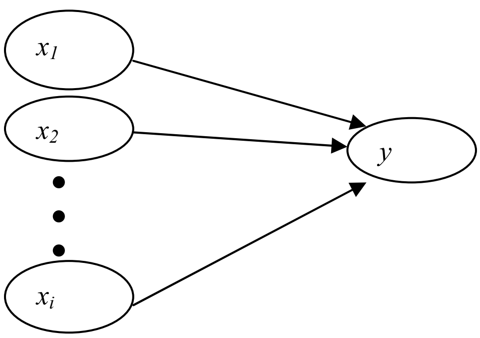

Figure 8. An FKM model of a multiple inputs single output (MISO) topology.

1. Randomly divide the data from the data set into two groups (Group A and Group B) such that the two groups contain equal number of data points. We denote x_{ij}^A and x_{ij}^B as the ith data points for jth input variable for Groups A and B respectively.

2. For each group, partition the input output spaces by fuzzy c-means clustering [23](or any appropriate clustering technique e.g. subtractive clustering[24]). We chose fuzzy c-means clustering so that comparison of performance of our approach to that of SY method can be made. We set the initial cluster size at 2. The centre v_i of the ith fuzzy cluster is given by:
$$v_i = \sum_{k=1}^{n} (\mu_{ik})^m x_k \;/\; \sum_{k=1}^{n} (\mu_{ik}), \quad 1 \le i \le c \tag{10}$$
where m is the adjustable weighting exponent which is set at 2 in our experiments; μ_{ik} is the degree of membership of x_i in kth cluster; and μ_{ik} is given by:
$$\mu_{ik} = 1 / \sum_{t=1}^{c} \left(\frac{\| x_i - c_k \|}{\| x_i - c_t \|} \right), \quad c_k = \frac{\sum_{i=1}^{n} \mu_{ik}^m \cdot x_i}{\sum_{i=1}^{n} \mu_{ik}^m} \tag{11}$$
where t is the iteration step.
We increment the number of clusters by 1 and recompute the cluster centres. The iteration stops when $\| U^{(t+1)} - U^{(t)} \| < \varepsilon$, where $U = \{\mu_{ik}\}$ matrix; and ε is a constant set at 0.00001 in this paper.

3. Partition the input spaces based on the optimal clusters obtained in step 2. Then construct fuzzy membership functions for each of the partitions such that they are ordered, consistent, and convex, and share common flanks.

4. For each of the input variables j, construct an FKM with a single input node N_j and an output node M as shown in Figure 7. That is, the number of FKM for each group should equal to the number of input variables. For clarity, we denote FKM_j^A and FKM_j^B as the FKMs for Group A and Group B respectively, with jth input node.

5. Based on the membership functions in Step 3, construct fuzzy rules for the FKMs.

6. Perform simulations for the FKMs, by inputting the data in Group A to FKM_j^B, and similarly inputting the data from Group B to FKM_j^A.

7. Calculate the regularity criterion [20] (RC for short) of the simulated outputs for each input variable j from Groups A and B, which is given by:

$$RC_j = \tfrac{1}{2}\{\sum_{i=1}^{k_A}(y_i^A - y_i^{AB})/k_A + \sum_{i=1}^{k_B}(y_i^B - y_i^{BA})/k_B\} \quad (12)$$

where k_A and k_B are the number of data points in the two groups A and B; y_i^A and y_i^B are the outputs of groups A and B respectively; and y_i^{AB} is the simulated output of FKM_j^A using Group B data as inputs, and y_i^{BA} is the simulated output of FKM_j^B using Group A data as input.

8. Pick the FKM having the smallest RC as being the winning FKM_j^* with the N_j^* input node as the winner for the jth input variable.

9. Compute simulated output y_i^{AA*} by performing FKM simulations using the winning FKM_j^* with input data from Group A. Similarly compute output y_i^{BB*} using FKM^* with Group B data.

10. Compute the new input data sets for Group A for the next potential input node N_j using the formula: $x_{ij}^A = y_i^A - y_i^{AA*}$ where $N_j \neq N_j^*$. Similarly compute the new input data set for Group B using the formula: $x_{ij}^B = y_i^B - y_i^{BB*}$.

11. Repeat from Step 2, but with the following modifications:

For the purpose of partitioning of the input-output spaces in Step 2, set the new input data sets for Groups A and B as computed in Step 10.

- In Step 4, for each of the loser input variables, take a copy of the winning FKM_i^* and add a loser node to form a MISO FKM as shown in Figure 10. The number of FKMs will be one less than that of the previous iteration.
- Note that in Step 6, the input data are those of the original data sets.
- In Step 8, if the smallest RC is greater than that in the previous iteration, the terminating condition has been reached. The iteration is terminated and the last winning node is discarded since it does not contribute towards reducing RC. If the smallest RC is still smaller than that in the previous iteration, the process continues until all the input variables have been added to the last winning FKM.

B. Construction of fuzzy rules based on expert knowledge

Single-antecedent fuzzy rules can be constructed using the various knowledge acquisition techniques available but in the context of construction of single-antecedent fuzzy rules. This issue is not discussed further in this paper.

However, where conventional multiple-antecedent fuzzy rules have been constructed from the expert knowledge, we can generate an input-output data set by computing the outputs for a set of randomly generated inputs. Extraction of single-antecedent fuzzy rules and construction of FKM can then proceed as given earlier in this section. This may be useful where a system may comprise of multiple FKMs linked together, and some of the FKMs need to acquire knowledge that has been expressed as conventional multiple-antecedent fuzzy rules. An example of a system consisting of multiple FKMs can be found in [25].

6. FKM examples and Experiments

We present two experimental modellings based on the data given in [20] for comparison of our approach and the SY method described in [20], which was based on the conventional multiple-antecedent fuzzy rules.

A. Simulation of human operation at a chemical plant

Consider a human operation of a chemical plant as illustrated by [20], where the plant produces a polymer by polymerisation of certain monomers. The control of the start-up of the plant is by a human. There are five sets of possible input candidates – monomer concentration, change of monomer concentration, monomer flow rate, and two temperature readings inside the plant (denoted as x_1, x_2, x_3, x_4 and x_5), and the control of the set point for monomer flow rate as the single output variable y. By "possible input candidates", we mean these are recoded observations which may or may not contribute towards the output y. There are 70 data points from the actual plant operation as given in [20], Dividing these into two groups gives us 35 data points per group.

Following our algorithm in Section V – A (2), first we constructed FKMs, each consisting of only one of the input variables, x_i and an output variable y, and an arc linking the input node to the output node. The arc represents a set of fuzzy rules which maps the input to the output spaces.

We found the clusters in the input and output spaces by using fuzzy c-means clustering, and then partitioning the input space. We used triangular membership functions for the inputs, and Takagi–Sugeno model of zero order (constant) for inference. Table 1 shows the cluster centres of the input and output spaces. Initially Table 1 consists of only column *Iteration #1* which corresponds to the FKMs with single input and single output (Column *Iteration* #2 corresponds to FKMs with two inputs and single outputs, and so on). From these clusters, we constructed the membership functions and the fuzzy rules to map the input to the output spaces. We then performed FKM simulation to obtain the output from which we compute the regularity criterion (RC) as in (9). This is tabulated in Table 2. As can be seen, for *Iteration #1,* input variable x_3 has the smallest RC and is therefore considered as the winner. The winning FKM was then selected and we added a loser node to it to form a new FKM for the next step in the iteration. The process terminates when either the RC for the current iteration is greater than that in the previous iteration, or when we run out of loser node to form a new FKM. In this experiment, the process stopped at *Iteration #3* when RC = 32,690 which is greater than RC = 20,563 in *Iteration #2.* The FKM in the final iteration is discarded since the RC is greater than the previous RC. As can be seen from Table 2, the final RC is 25,563 for our FKM model as compared with 36,950 for SY method. The result shows that our method performed better than SY, with an RC 30% less than that of SY.

Table 1. Cluster canters of input - output spaces of the operation of the chemical plant

Iteration #1		**Iteration #2**		**Iteration #3**	
x_1	y	x_1	y	x_1	y
4.60	6,716	4.60	275		
5.59	3,985	5.14	8	N/A	
6.22	1,043	6.49	- 305		
x_2	y	x_2	y	x_2	y
- 0.10	1,041	- 0.22	- 54	0.06	14
- 0.04	3,979	0.04	77	- 0.01	- 190
- 0.01	6,709			- 0.04	185
x_3	y	x_3	y	x_3	y
961	1,031				
3,914	4,045	N/A		N/A	
6,673	6,744				
x_4	y	x_4	y	x_4	y
- 0.09	1,020	- 0.06	- 72	- 0.08	- 188
- 0.09	6,700	- 0.04	245	0.09	186
- 0.01	3,980			- 0.03	14
x_5	y	x_5	y	x_5	y
- 0.00	6,701	0.01	- 72	0.01	14
0.01	3,988	0.03	247	0.03	185
0.06	1,041			- 0.03	- 188

Table 2. Simulations results using fkm and sy methods

	Input variable	**RC**	
		FKM	SY
Iteration #1	x_1	606,023	602,715
	x_2	6,215,368	6,077,539
	x_3	**41,053**	**60,756**
	x_4	11,828,720	6,663,660
	x_5	8,329,532	5,570,199
Iteration #2	$x_3 + x_1$	**20,563**	46,178
	$x_3 + x_2$	39,695	**41,418**
	$x_3 + x_4$	50,590	60,124
	$x_3 + x_5$	68,689	60,277
Iteration #3	$x_3 + x_1 + x_2$	**32,690**	**36,950**
	$x_3 + x_1 + x_4$	52,728	N/A
	$x_3 + x_1 + x_5$	36,206	41,846

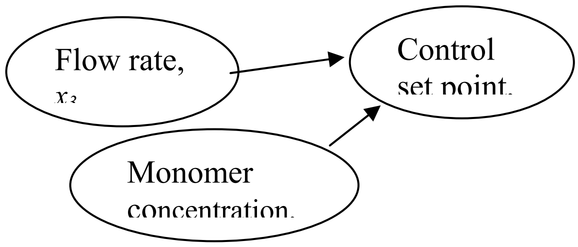

Figure 9. Dual input single output FKM model for Polymerisation Chemical Plant.

Our final winning FKM therefore consists of two input variables $x_3 + x_1$. The FKM model for the chemical plant is a DISO model, as shown in Figure 9.

In FKM, the rules are single-antecedent fuzzy rules. As such, we avoid rule transparency problem. This is apparent from the rules used in the SY method and FKM.

The fuzzy rules of the SY method are as follows:

1) If x_1 is more or less BIG, and x_2 is not INCREASED, and x_3 is SMALL, then y is SMALL or MEDIUM SMALL.
2) If x_1 is more or less MEDIUM, and x_2 is DECREASED, and x_3 is SMALL or MEDIUM SMALL, then y is MEDIUM SMALL.
3) If x_1 is MEDIUM, and x_2 shows NO CHANGE, and x_3 is MEDIUM SMALL or MEDIUM, then y is MEDIUM.
4) If x_1 is more or less MEDIUM, and x_2 is ANY VALUE, and x_3 is MEDIUM, then y is MEDIUM or MEDIUM BIG.
5) If x_1 is more or less SMALL, and x_2 is very

INCREASED, and x_3 is MEDIUM BIG, then y is BIG.
6) If x_1 is more or less SMALL, and x_2 is sort of INCREASED, and x_3 is BIG, then y is very BIG.

As can been seen, the rules are difficult to comprehend. In fact [20] have resorted to fine-tuning the rules by implementing additional fuzzy labels such as “more or less”, “not”, “very”, “more than”, etc. Our method did not need such fine tuning.

The rules used in FKM are as follows:

1) If x_3 is LOW, then y_{x3} = 1031
2) If x_3 is MEDIUM, then y_{x3} = 4045
3) If x_3 is HIGH, then y_{x3} = 6744
4) If x_1 is LOW, then y_{x1} = 275
5) If x_1 is MEDIUM, then y_{x1} = 8
6) If x_1 is HIGH, then Y_{x1} = -305

The first three rules are implemented in the arc between node *flow rate* and node *control set point,* and the next three rules between *monomer concentration* and *control set point.* As can be seen in Figure 9, y_{x3} and y_{x1} in the above rules are the outputs from x_3 and x_1 respectively. These outputs are aggregated to form the final output y. As can be seen, the above rules are easy to understand, especially in conjunction with the FKM in Figure 9.

B. Simulation of a nonlinear system

Consider a nonlinear function with two inputs x_1 and x_2, and a single output y:

$$y=(1+x_1^{-2}+x_2^{-1.5})^2 \tag{13}$$

M. Sugeno and T. Yasukawa [20] used this formula to generate a set of 50 input-output data points with two extra dummy input variables x_3 and x_4 as random noise in the system, where $1 \le x_1, x_2, x_3, x_4 \le 5$. We used the same dataset available from [20] for our FKM simulation. That is, we have four input candidates, $x_1 - x_4$, and have to find the actual inputs that affect the output y.

As in the first experiment, we divided the dataset into two groups. At each iteration, we added a loser node to the winning FKM, partitioned the input-output spaces for the new node, and formulated the fuzzy rules. In this experiment, the iteration terminated at third iteration when the RC was greater than that at second iteration.

Table 3 shows the cluster centres of the input-output spaces, and Table 4 shows that both our approach and SY method identified correctly the input variables x_1 and x_2. However, our approach was better than SY method in that the final RC of our approach was 0.272 whereas that of SY was 0.424. Our final FKM model is as shown in Figure 10. In other words, x_1 is the winning variable from iteration #1 and x_2 is the winning variable from iteration #2. These are highlighted in bold in Table 3.

Table 3. Cluster centers of input - output spaces of the non-linear function (13)

Iteration #1		Iteration #2		Iteration #3	
x_1	y				
1.42	**4.09**	N/A		N/A	
1.88	**2.58**				
3.93	**2.12**				
x_2	y	x_2	y		
1.76	2.91	1.76	**0.43**	N/A	
4.23	2.01	4.12	**-0.30**		
x_3	y	x_3	y	x_3	y
1.57	2.59	2.09	-0.01	1.54	0.38
2.50	2.20	4.22	-0.02	2.78	0.04
2.82	4.68			4.42	0.30
4.24	3.41				
4.47	2.05				
x_4	y	x_4	y	x_4	y
1.85	1.64	1.54	0.71	1.28	0.36
2.06	4.58	2.39	-0.12	2.42	0.33
3.59	1.69	3.80	-0.22	3.58	0.48
4.03	3.34			4.01	-0.63

Table 4. Simulation results of fkm and sy methods on non-linear function (13)

	Input variable	RC	
		FKM	SY
Iteration #1	x_1	**0.817**	**0.630**
	x_2	0.845	0.863
	x_3	2.001	0.830
	x_4	1.556	0.937
Iteration #2	$x_1 + x_2$	**0.272**	**0.424**
	$x_1 + x_3$	0.560	0.571
	$x_1 + x_4$	0.652	0.583
Iteration #3	$x_1 + x_2 + x_3$	**0.282**	**0.483**
	$x_1 + x_2 + x_4$	0.294	0.493

Listed below are the fuzzy rules in SY method:

1) If x_1 is more than MEDIUM and x_2 is more MEDIUM, then y is SMALL, $\partial y/\partial x_1$ is sort of

NEGATIVE, and $\partial y/\partial x_2$ is sort of NEGATIVE.
2) If x_1 is not SMALL but less than MEDIUM BIG and x_2 is not SMALL but less than MEDIUM BIG, then y is MEDIUM SMALL, $\partial y/\partial x_1$ is sort of NEGATIVE, and $\partial y/\partial x_2$ is sort of NEGATIVE.
3) If x_1 is not SMALL but less than MEDIUM BIG and x_2 is more or less MEDIUM BIG, then y is more or less MEDIUM SMALL and MEDIUM, $\partial y/\partial x_1$ is sort of NEGATIVE, and $\partial y/\partial x_2$ is sort of NEGATIVE.
4) If x_1 is less than MEDIUM and x_2 is less than MEDIUM BIG, then y is MEDIUM and more or less MEDIUM BIG, $\partial y/\partial x_1$ is sort of NEGATIVE, and $\partial y/\partial x_2$ sort of NEGATIVE.
5) If x_1 is more or less MEDIUM SMALL or MEDIUM and x_2 is more or less SMALL, then y is MEDIUM BIG, $\partial y/\partial x_1$ is sort of NEGATIVE, and $\partial y/\partial x_2$ is NEGATIVE.
6) If $x=1$ is SMALL and x_2 is more or less SMALL or MEDIUM SMALL, then y is BIG, $\partial y/\partial x_1$ is VERY NEGATIVE, and $\partial y/\partial x_2$ is very NEGATIVE.

$\partial y/\partial x_1$ and $\partial y/\partial x_2$ are modifiers of position gradient type used in SY method. Further description can be found in [20].

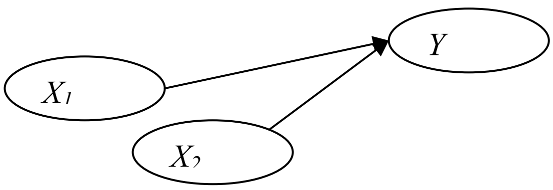

Figure 10. FKM model for non-linear system (13).

The winning variables x_1 and x_2 as highlighted in bold in Table 3 form the basis of construction of the fuzzy rules of our FKM model. These fuzzy rules are as follows:

1) If x_1 is LOW then y = 4.09
2) If x_1 is MEDIUM then y = 2.58
3) If x_1 is HIGH then y = 2.12
4) If x_2 is LOW then y = 0.43
5) If x_2 is HIGH then y = -0.30

The first three rules are implemented in the arc between nodes X_1 and Y in the FKM (Figure 10), whereas the last two rules are implemented in the arc between X_2 and Y. Although there are only two input variables in this experiment, the antecedent rules in our FKM model not only perform better than those in SY method, but also are much easier to comprehend.

7. Conclusion

The conventional linguistic conjunctive multiple-antecedent form of fuzzy rules has been known to cause transparency and complexity problems in fuzzy systems and is not suitable for implementation in FCM.

We have shown that single-antecedent fuzzy rules can approximate a polynomial function and therefore a continuous function. We have shown how single-antecedent fuzzy rules can be implemented in place of multi-antecedent rules. This eliminates the potential rule explosion problem and significantly simplifies the logic reasoning processes. We have demonstrated this using FKM. Where the multi-antecedent parts are large and the number of rules is huge, our approach has a clear advantage in terms of computational overheads and maintenance.

We have also shown that the linguistic labels used in single-antecedent fuzzy rules are simpler to comprehend. We observe that generally when fuzzy rules are data driven, accuracy of approximation almost always take priority over semantic clarity of labelling of these rules. Hence the linguistic labels become obscured and more ambiguous. In an FKM environment, each set of single-antecedent fuzzy rules relates specifically to each input and output variables, thus eliminating any ambiguity.

The simulation experiments clearly show that our approach to rule construction is simpler and enhances significant performance improvement over methods such as SY method where multiple-antecedent rules are used.

8. References

[1] R. Axelrod, *Structure of Decision: The Cognitive Maps of Political Elites,* Princeton, New Jersey: Princeton University Press, 1976, pp. 404.
[2] B. Kosko, "Fuzzy cognitive maps," *Int. Journal of man-machine studies*, vol. 24, pp. 66-75, 1986.
[3] B. Kosko, *Fuzzy Engineering*. New Jersey, USA: Prentice-Hall, Inc., 1997.
[4] K. Perusich, "Fuzzy cognitive maps for policy analysis," *International Symposium on Technology and Society (ISTAS'96)*, Purdue University, pp. 369-373, 1996.
[5] R. Taber, "Fuzzy cognitive maps model social systems," *AI Expert*, vol. 9, pp. 18-23, 1994.
[6] M. Hagiwara, "Extended fuzzy cognitive maps," *1st IEEE Int. Conf. on Fuzzy Systems*, San Diego, CA, USA, pp. 795-801, 1992.
[7] R. Satur and Z.-Q. Liu, "A contextual fuzzy cognitive map framework for geographic information systems,"

IEEE Trans. on Fuzzy Systems, vol. 7, no.5, pp. 481-494, 1999.

[8] J. P. Carvalho and J. A. B. Tomé, "Rule based fuzzy cognitive maps: fuzzy causal relations," *Int. Conf. on Computational Intelligence for Modelling, Control and Automation - CIMCA'99*, Vienna, Austria, pp. 1-5, 1999.

[9] Y. Miao and Z.-Q. Liu, "On causal inference in fuzzy cognitive maps," *IEEE Transactions on Fuzzy Systems*, vol. 8 ,no.1, pp. 107-119., 2000.

[10] J. P. Carvalho and J. A. B. Tomé, "Issues on the stability of fuzzy cognitive maps and rule-based fuzzy cognitive maps," *NAFIPS-FLINT 2002, North American Fuzzy Information Processing Society*, New Orleans, LA, pp. 1-8, 2002.

[11] B. Kosko, *Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence*, New Jersey, USA: Prentice-Hall, Inc., 1992.

[12] J. P. Carvalho and J. A. Tomé, "Rule based fuzzy cognitive maps — expressing time in qualitative system dynamics," *10th IEEE Int. Conf. on Fuzzy Systems*, Melbourne, Australia, pp. 280-283, 2001.

[13] M. S. Khan, S. W. Khor and J. Payakpate, "Fuzzy knowledge representation for decision support," *5th Int. Conf. on Knowledge Based Computer Systems (KBCS 2004)*, Hyderabad, India, pp. 186-195, 2004.

[14] S. W. Khor, M. S. Khan and C. C. Fung, "Fuzzy modelling using reduced rule sets," *Postgraduate Electrical Engineering and Computing Symposium (PEECS 2004)*, Perth, Australia, pp. 169-174, 2004.

[15] H. Ying, "Sufficient conditions on uniform approximation of multivariate functions by general Takagi–Sugeno fuzzy systems with linear rule consequent," *IEEE Trans. on Systems, Man, and Cybernetics - Part A*, vol. 28, no.4, pp. 515-520, 1998.

[16] X.-J. Zeng and M. G. Singh, "Approximation theory of fuzzy systems - MIMO case," *IEEE Trans. Fuzzy Systems*, vol. 3, no.2, pp. 219-235, 1995.

[17] J. L. Castro and M. Delgado, "Fuzzy systems with defuzzification are universal approximators," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 26, no.1, pp. 149-152, 1996.

[18] B. Kosko, "Fuzzy systems as universal approximators," *IEEE Int. Conf. Fuzzy Systems*, San Diego, CA, pp. 1153-1162, 1992.

[19] W. Rudin, *Principles of Mathematical Analysis*, New York: McGraw-Hill, 1964.

[20] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling," *IEEE Transactions on Fuzzy Systems*, vol. 1, no.1, pp. 7-29, 1993.

[21] K. W. Wong, L. T. Kóczy, T. D. Gedeon, A. Chong and D. Tikk, "Improvement of the cluster searching algorithm in Sugeno and Yasukawa's qualitative modeling appro," *Lecture Notes in Computer Science*, vol. 2206, pp. 536-549, 2001.

[22] S. W. Khor, "An experimental study of a fuzzy rules reduction method using the Sugeno and Yasukawa's qualitative model," *Int. Conf. on Fuzzy Information Processing: Theories and Applications (FIP 2003)*, Tsinghua University, Beijing, China, pp. 461-466, 2003.

[23] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Hardbound: Plenum Publishers, 1981.

[24] S. Chiu, "Fuzzy model identification based on cluster estimat," *Intelligent Fuzzy Systems*, vol. 2, no.3, pp. 267-278, 1994.

[25] M. S. Khan and S. W. Khor, "A fuzzy modelling framework for stock market analysis," *5th Academy of Business and Information Technology*, Pennsylvania, USA, 2005.