Evolutionary learning of rule premises for fuzzy modelling

N. Xiong
Published online: 26 Nov 2010.

To cite this article: N. Xiong (2001) Evolutionary learning of rule premises for fuzzy modelling, International Journal of Systems Science, 32:9, 1109-1118, DOI: 10.1080/00207720010015735

To link to this article: http://dx.doi.org/10.1080/00207720010015735
Evolutionary learning of rule premises for fuzzy modelling

N. Xiong†

The task of fuzzy modelling involves specification of rule antecedents and determination of their consequent counterparts. Rule premises appear here a critical issue since they determine the structure of a rule base. This paper proposes a new approach to extracting fuzzy rules from training examples by means of genetic-based premise learning. In order to construct a 'parsimonious' fuzzy model with high generalization ability, general premise structure allowing incomplete compositions of input variables as well as OR connectives of linguistic terms is considered. A genetic algorithm is utilized to optimize both the premise structure of rules and fuzzy set membership functions at the same time. Determination of rule conclusions is nested in the premise learning, where consequences of individual rules are determined under fixed preconditions. The proposed method was applied to the well-known gas furnace data of Box and Jenkins to show its validity and to compare its performance with those of other works.

1. Introduction

A fuzzy model consists of a set of fuzzy if–then rules to describe the input–output relation of a system. The significance of fuzzy models arises because there exist in the real world such problems that lack an exact mathematical description and therefore can hardly be solved by conventional methods. The task of fuzzy modelling involves specification of rule antecedents and determination of their consequent counterparts. Rule premises appear here a critical issue as they determine the structure of a rule base. Recently fuzzy clustering techniques have been widely used to generate fuzzy rules automatically from training data, where it is required to identify a particular membership function for every input variable in a rule condition (Sugeno and Yasukawa 1993, Hwang and Woo 1995, Kim et al. 1997, Klauwonn and Kruse 1997, Emami et al. 1998). In this way the fuzzy models generated exploit local fuzzy sets pertaining to individual rules rather than the global fuzzy sets used by all rules. We cannot ensure that the local membership functions have any semantic values, so that a linguistic interpretation of the rules is difficult.

For easier interpretation we advocate building fuzzy models with global fuzzy sets, which can be used by all rules and to which some qualitative meanings such as ‘large’ and ‘small’ can be assigned. The identification of the premise structure is thus equivalent to selecting appropriate logic connectives of global (input) fuzzy sets. Utilization of all AND combinations of input fuzzy sets as rule conditions provides an easy way to define model structure for simple problems with a low number of inputs (Wang and Mendel 1992, Liska and Melsheimer 1994, Jin and Jiang 1995, Preuß and Ockel 1995, Cho et al. 1998, Rahmoun and Benmohamed 1998). However, this scheme could lead to combinatorial explosion of the rule number when the system to be modelled has a high input dimension. In such cases, another premise structure than those canonical AND connectives is needed to reduce the number of rules required.

This paper presents a new method to evolve rule premises by genetic algorithms (GAs). We introduce here a general premise structure allowing not only incomplete composition of input variables but also OR connectives of linguistic terms, so that a high generalization ability can be achieved by an individual rule. The upper limit of the rule number is predetermined by man in advance. It can be considered as an estimation of the sufficient amount of rules to achieve a satisfactory accuracy. During the running of GA the actual rule number can be adjusted automatically within this specified limit. The modelling procedure for a fuzzy system consists of
extern loop and intern loop, as illustrated in figure 1. In the external loop a GA is utilized to search in the combinatorial space for the optimal structure of premises as well as to optimize parameters of fuzzy set membership functions simultaneously. The internal loop serves for fitness evaluation for premise learning and it is responsible for determining the consequences of individual rules under fixed preconditions.

The basic philosophy behind the proposed modelling approach is to obtain fuzzy models exhibiting not only good predictive properties but also strong descriptive capabilities. The rule conditions learned in this paper are composed of appropriate logic connectives of global fuzzy sets, from which a linguistic description of the input situation is possible. Predictive properties of fuzzy models are ensured by premise learning based on a GA that searches for optimal premise structure and input membership functions concurrently. Owing to the introduction of rule conditions of a general form as well as the selection of important coverage of the input domain, a compact rule base can be expected from our method. Certainly, a compact fuzzy model with a small number of rules has a high generalization ability and is also easy for human inspection and understanding.

The remainder of this paper is organized as follows. In §2 a brief description is given of the general rule premises considered in this paper together with their advantages in knowledge representation. An approach to automatically learning such premises using GA is proposed in the following section. In §4 a numerical example of fuzzy modelling the gas furnace data of Box and Jenkins (1970) is provided to demonstrate the effectiveness of the proposed method. Finally concluding remarks are presented in §5.

2. General premises versus elementary premises
Suppose that a fuzzy model with $X = (x_1, x_2, \ldots, x_n)$ as its inputs and $y$ as its output. Each input $x_i$ ($i = 1, \ldots, n$) has $q[i]$ linguistic terms denoted as $A(i, 1), A(i, 2), \ldots, A(i, q[i])$. $p(\ )$ is an integer function mapping from $\{1, 2, \ldots, s(s \leq n)\}$ to $\{1, 2, \ldots, n\}$ satisfying $\forall x \neq y, p(x) \neq p(y)$. A general rule premise is formulated as follows.

Definition 1: A general rule premise is a compound fuzzy proposition of the form as

\[
\begin{align*}
  x_{p(1)} &= \bigcup_{j \in D(1)} A(p(1), j) \\
  x_{p(2)} &= \bigcup_{j \in D(2)} A(p(2), j) \\
  \ldots \ldots & \ldots \\
  x_{p(s)} &= \bigcup_{j \in D(s)} A(p(s), j), \tag{1}
\end{align*}
\]

where $D(k) \subset \{1, 2, \ldots, q[p(k)]\}, \quad k \in \{1, 2, \ldots, s\}$. 

Clearly the premise formulated by (1) can be regarded as a conjunction of simple propositions, each of which corresponds to OR connectives of fuzzy sets defined for the input variable involved in it. If such a premise includes simple propositions for all inputs (e.g. $s = n$), we say that this condition has a complete structure, otherwise its structure is incomplete.

Definition 2: The premise expressed in (1) is an elementary rule premise if the following conditions hold
(i) $s = n$. 
(ii) $\forall k \in \{1, 2, \ldots, n\}$, $\|D(k)\| = 1$. 

Elementary rule premises defined above have complete structure and contain only one linguistic term for every input variable. They can be regarded as 'degen-
eration’ from the general rule premises in (1). Using canonical AND connectives of linguistic terms, elementary rule premises are especially suitable for describing simple systems with low input dimensions.

For fuzzy modelling of complex processes with high input dimensions, general rule premises are preferable, since they can achieve larger coverage of the input domain compared with elementary premises. The main purpose of this paper is to find such premises optimally so that a compact rule base is possible. A rule antecedent with incomplete structure or containing OR connectives of linguistic terms can replace a set of related elementary premises, as illustrated in the following two examples.

**Example 1:** The antecedent ‘(x1 = NZ or PZ) and (x2 = NZ or PZ)’ shown in figure 2 covers four elementary premises as follows:

(i) if (x1 = NZ) and (x2 = NZ);
(ii) if (x1 = NZ) and (x2 = PZ);
(iii) if (x1 = PZ) and (x2 = NZ);
(iv) if (x1 = PZ) and (x2 = PZ).

**Example 2:** The incompletely structured antecedent ‘(x1: don’t care) and (x2 = Z)’ shown in figure 3 covers the following group of elementary premises:

(i) if (x1 = N) and (x2 = Z);
(ii) if (x1 = NZ) and (x2 = Z);
(iii) if (x1 = Z) and (x2 = Z);
(iv) if (x1 = PZ) and (x2 = Z);
(v) if (x1 = P) and (x2 = Z).

Rule interestingness is an important aspect of knowledge representation for neurofuzzy systems (Wittmann et al. 1999). Informally speaking, the more general the antecedent description and the more specific the consequent part are, the more interesting and informative the rule is. The premises in the above two examples are more interesting than the elementary premises covered by them as the input situations covered by the elementary premises are relatively much narrower. In this view, the introduction of the general premise structure as explained in this section helps to enhance the interestingness of a fuzzy rule base to be modelled.

3. Genetic learning of rule premises

GAs (Holland 1975, Goldberg 1989) are global search algorithms that emulate the mechanics of natural genetics and selection. Based on probabilistic decisions they exploit historic information to guide the search for new points in the problem space with expected improvement in performances. In the genetic search a constant population size is always maintained. An individual in the population encodes a possible solution to the problem into a string, which is analogous to a chromosome in nature. At each iteration step, new strings are created by applying genetic operators on selected parents for recombination. Coding scheme, genetic operators (reproduction, crossover and mutation) and fitness function are key points for the GA to optimize the structure of rule premises and input membership functions at the same time.

3.1. Binary coding of premise structure

From definition 1, we can see that the surface structure of a rule condition is characterized by sets \( D(k) \subset \{1, 2, \ldots, q[p(k)]\} \). This fact suggests that a binary code be a suitable scheme for representing premise structure of such a form, as inclusion or exclusion of an integer in the sets \( D(k) \) can be declared binary. For input variable \( x_i(i = 1, 2, \ldots, n) \) with \( q[i] \) linguistic terms, a segment consisting of \( q[i] \) binary bits is required to encode the conditional composition for this variable.
Every bit of the segment corresponds to a linguistic term with bit ‘1’ for presence and bit ‘0’ for absence of its fuzzy set in forming the condition. For example, assume that \(x_i\) has three linguistic terms \{low, middle, high\}, the segments ‘010’ and ‘100’ correspond to the conditions ‘\(x_i = \text{middle}\)’ and ‘\(x_i = \text{low}\)’ respectively. Similarly, the condition with OR operation ‘\(x_i = \text{middle or high}\)’ is represented by the segment ‘011’. All-one segment ‘111’ is adopted in this paper to represent the wildcard of ‘don’t care’, meaning that the corresponding input variable is not considered in the rule premise. By combining together the binary segments for all individual inputs, the whole surface structure of a rule condition can be encoded into the group \(P = (S(x_1), S(x_2), \ldots, S(x_n))\), where \(S(x_i)(i = 1, \ldots, n)\) denotes the segment of the condition description for input variable \(x_i\).

**Example 3:** Assume four input variables \(x_1, x_2, x_3\) and \(x_4\) with \(x_i = \{\text{low, middle, high}\} (i = 1, \ldots, 4)\). The structure of a rule premise: \((x_1 = \text{low or high})\) and \((x_3 = \text{middle})\) and \((x_4 = \text{middle or high})\) corresponds to the following binary coding:

\[
S(x_1) = 101, \quad S(x_2) = 111, \quad S(x_3) = 010, \quad S(x_4) = 011,
\]

\[
P = [S(x_1), \quad S(x_2), \quad S(x_3), \quad S(x_4)]
\]

\[
= [101\, 111\, 010\, 011].
\]

Further, the coding of premise structure of the whole rule base is realized through merging binary groups of all individual rule premises in a head-to-tail manner. Let \(L_m\) be the maximal number of rules allowed to appear in the rule base, the binary code \(CB\) for the premise structure of the whole system is written as

\[
CB = \{P(1), P(2), \ldots, P(L_m)\},
\]

where \(P(j)(j = 1, 2, \ldots, L_m)\) indicates the binary group for the premise structure of the \(j\)th rule.

### 3.1.1. Variable sized rule base implicated

It is worthy noting that the following two cases in a bit group of the binary code lead to an invalid rule premise encoded.

(a) All the bits in the group are equal to one, meaning that all input variables are neglected in the premise.

(b) The group contains an all-zero segment. Its corresponding input variable thus takes no linguistic term in the premise, resulting in an empty fuzzy set for the condition of that input.

Rules with invalid premises are meaningless or play no role in the fuzzy reasoning. Therefore they should be removed from the knowledge base. If all the binary groups \(P(j)\) in (2) correspond to valid premises, the rule base has exactly \(L_m\) rules in it. Otherwise the size of the rule base can be reduced according to invalid premises detected in (2). It is clear that the exact size of the rule base is not determined by man in advance. Rather the actual rule number is implicated in the binary code \(CB\), which is to be learned by GA. In this view, we claim that adjustment of the size of the rule set is possible within the constraint of prescribed maximal rule amount.

### 3.2. Integer coding of input membership functions

Triangular or trapezoid formed fuzzy sets are adopted in this paper. To achieve an optimal interface (de Oliveira 1999) we require that the sum of membership values for every input variable be always equal to one, so that only certain end points (also peaks) of membership functions need to be tuned (e.g. see the fuzzy sets in figure 4). Let the parameter for an end point be mapped by an integer \(N\) in the interval \(\{0, 1, \ldots, N_{\text{max}}\}\), the relationship between the parameter value \(V\) and the integer \(N\) is as follows:

\[
V = V_{\text{min}} + \frac{N}{N_{\text{max}}} (V_{\text{max}} - V_{\text{min}}),
\]

where \(V_{\text{min}}\) and \(V_{\text{max}}\) are two extreme limits of the end point. For fuzzy sets of an input variable, the distribution of their membership functions is determined by several important end points and each of the end points can be quantized by an integer. The arrangement of such integers in order results in an integer chain depicting the fuzzy partition of that input. For instance, the exact definition of the fuzzy sets in figure 4 is in fact transformed to specification of positions of the four end points. If we quantize the possible range of the positions from 0 to 100, the six membership functions can be characterized by the integer chain \(M = (15, 25, 45, 70)\). The elements in the chain \(M\) are ordered from small to large with correspondence to end points from left to right.

Further, the integer chains for membership functions of all input variables are merged together to produce an integer code to represent information of all input fuzzy sets as a whole. Suppose that there are \(n\) input variables in the system and \(M(i)(i = 1, 2, \ldots, n)\) is the integer chain for membership functions of the \(i\)th input, the
integer code CI for input membership functions of the whole system is given by
\[ CI = [M(1), M(2), \ldots, M(n)]. \quad (4) \]

3.3. Hybrid coding of rule conditions

Surface structure and input fuzzy sets are two essential aspects to describe the rule conditions of a fuzzy knowledge base. Because of the inherent relationship between the premise structure and the membership functions, it is preferable that both of them be optimized simultaneously. For this purpose, a hybrid string consisting of binary code in (2) and integer code in (4) as its two substrings is suggested here. Figure 5 illustrates the constitution of such a hybrid string proposed. The substring on the left is the binary code for premise structure of the rule base. \( L_m \) denotes the upper limit of the total number of rules in the rule base and \( P(j)(j = 1, \ldots, L_m) \) is the binary group corresponding to the premise structure of the \( j \)th rule. The substring on the right is the integer code representing membership functions of all input fuzzy sets, on which the surface structure of rule conditions depends. \( M(1), M(2), \ldots, M(n) \) indicate the integer chains for membership functions of input variables \( 1, 2, \ldots, n \) respectively. Such a hybrid string is considered as a chromosome in the GA population, which corresponds to a possible specification of the fuzzy conditions of a fuzzy model. Through evolutionary process based on genetic operators the quality of chromosomes in the population can be gradually improved.

3.4. Fitness evaluation

Evaluation of a (hybrid) string HS is based upon the performance of the fuzzy model resulting from it. On the one hand we require the model to be built to predict system behaviour as accurately as possible. On the other hand a compact knowledge base containing fewer rules is more desirable and interesting. According to the above two criteria, the cost function for a string HS is constructed as follows:
\[ \cos t(\text{HS}) = \sum_{k=1}^{N_f} (\hat{y}_i(k) - y(k))^2 + \beta R_S(\text{HS}). \quad (5) \]

Here \( R_S(\text{HS}) \) indicates the actual real number of the fuzzy model resulted from the string HS and \( \beta \) is a coefficient determined by man. \( \hat{y}_i(k) \) denotes the model’s estimate of the real output \( y(k) \). \( N_f \) is the number of the training data. Because the size of the rule base is now directly incorporated into the evaluation function, the GA searches for solutions with not only best accuracy but also minimal complexity.

To acquire the model output, the consequences of fuzzy rules must first be identified according to their pre-conditions. This is the reason why the procedure of conclusion determination is nested in premise optimization. In this paper we adopt linear regression equations as rule conclusions (Takagi and Sugeno 1985), and the parameters of consequent functions are identified using the weighted least-square estimation (see appendix A). In other cases, for example for fuzzy classifiers, the conclusion under a fixed antecedent can be selected through a heuristic procedure (Nozaki et al. 1996).

3.5. Genetic operations

By the operation of crossover, parent strings mix and exchange their attributes through a random process with expected improvement in fitness in the next generation. Owing to the distinct natures of the two substrings, it is preferable that the attributes in both substrings be mixed and exchanged separately. For this purpose a special three-point crossover is recommended here. One break point of this operation is fixed to be the splitting point between the two substrings, and the other two break points can be randomly selected within the two substrings respectively. At break points the parents’ bits are alternatively passed on to the offspring. This means that the offspring obtain bits from one of the parents until a break point is encountered, at which point they switch and take bits from the other parent.

Example 4: Consider two chromosomes in the following:
\[ X_1 = (b_1^1, b_1^2, b_1^3, b_1^4, b_1^5, b_1^6, b_1^7, b_1^8, b_1^9, b_1^{10}, b_1^{11}, b_1^{12}), c_1^1, c_1^2, c_1^3, c_1^4, c_1^5, c_1^6) \]
\[ X_2 = (b_2^1, b_2^2, b_2^3, b_2^4, b_2^5, b_2^6, b_2^7, b_2^8, b_2^9, b_2^{10}, b_2^{11}, b_2^{12}), c_2^1, c_2^2, c_2^3, c_2^4, c_2^5, c_2^6) \]

Both \( X_1 \) and \( X_2 \) consist of two substrings \( (b_i^1, b_i^2, \ldots, b_i^{12}), (c_i^1, c_i^2, \ldots, c_i^6)(i = 1, 2) \) representing...
the premise structure and the parameters of input fuzzy sets respectively. The position between \(b_i^1\) and \(c_i^1\) is the splitting point between two sub-strings. Selecting the other two break points for the crossover operator as the position between \(b_i^2\) and \(b_i^3\) and the position between \(c_i^4\), \(c_i^5\), we obtain the offspring as follows:

\[
Y_1 = (b_1^1, b_1^2, b_1^3, b_1^4, b_1^5, b_2^6, b_2^7, b_2^8, b_2^9, b_2^{10}, b_2^{11}, b_2^{12}) \\
\quad \quad \quad \quad c_1^1, c_1^2, c_1^3, c_1^4, c_1^5, c_2^6
\]

\[
Y_2 = (b_1^1, b_1^2, b_2^3, b_2^4, b_2^5, b_2^6, b_1^7, b_1^8, b_1^9, b_1^{10}, b_1^{11}, b_1^{12}) \\
\quad \quad \quad \quad c_2^1, c_2^2, c_2^3, c_2^4, c_2^5, c_1^6
\]

Clearly this three-point crossover used here is equivalent to two one-point crossovers operating on both sub-strings separately.

Mutation is a random alteration of a bit in a string so as to increase the variability of population. Because of the distinct sub-strings used, different mutation schemes are needed to suit their purposes.

(a) Membership function mutation. Since parameters of input membership functions are essentially continuous, a small mutation with high probability is more meaningful. Therefore it is so designed that each element in the integer substrings for membership functions undergo a disturbance. The magnitude of this disturbance is determined by a Gaussian random variable \(N(0, \sigma)\) with mean 0 and variance \(\sigma\).

(b) Premise structure mutation. Bit mutation is applied on the binary substrings codifying the surface structure of rule conditions. This is a random operation that occasionally occurs with a small probability (typically 0.01–0.05). Each bit of the binary code is flipped if a probability test is satisfied, that is a randomly generated real number is smaller than the prespecified probability.

Parent selection is a routine emulating the mechanism of survival of the better fitness in nature. It is expected that a better hybrid string will produce a higher number of offspring and thus has a higher chance of surviving in the subsequent generation. The selection rate prob (HS) of a hybrid string HS is determined by

\[
\text{prob (HS)} = \frac{C - \text{cost (HS)}}{(L_p - 1)C},
\]

where \(L_p\) denotes the population size and \(C\) is the sum of cost values of all individuals in the population. Finally a new population is generated as follows

(i) A set of offspring is created by selecting parents from current generation and applying genetic operators for recombination. Each integer chain in the offspring then undergoes a reordering opera-

(ii) The individuals in the offspring set are evaluated in terms of accuracy and complexity of the associated fuzzy model using the cost function (5).

(iii) The best \(L_p\) individuals from the current population and the offspring set are chosen to form the next generation (\(L_p\) is the population size).

4. Numerical example

The proposed approach was applied to a well-known problem of modelling a gas furnace system introduced by Box and Jenkins (1970). This data set consists of 296 input–output measurements from a gas furnace system with a single input \(u\) being the gas flow rate and a single output \(y\) being the \(\text{CO}_2\) concentration in the outlet gas. The sampling interval is 9 s. To compare with other conventional models, \(u(k), u(k - 1), u(k - 2), y(k - 1), y(k - 2)\) and \(y(k - 3)\) were chosen as input variables of the fuzzy model. Three fuzzy sets (negative, zero and positive) are used to depict the gas flow rate and another three fuzzy sets (low, middle and high) correspond to the \(\text{CO}_2\) concentration \(y(k)\). It is required that the sum of membership values for every variable be always equal to unity, so that only four critical points (\(a_1\) and \(a_2\) in figure 6, and \(b_1\) and \(b_2\) in figure 7) are to be adapted. A GA was used to learn the structure of rule premises as well as to optimize the parameters \(a_1, a_2, b_1\) and \(b_2\) of the input fuzzy sets simultaneously.

Regarding the size of the rule base, we had no idea at first about how many rules should be included. Only an upper limit of the rule number was determined as ten, meaning that ten rules were supposed to be sufficient to achieve a desirable modelling accuracy. The possible rule conditions were evolved by GA which searched for solutions by minimizing the cost function (5). That is, not only modelling accuracy but also model complexity were optimized by GA at the same time.

The parameters for the GA used in our experiments are as follows: population size, 100; crossover rate, 0.867; mutation rate (for the premise structure), 0.033.

![Figure 6. Fuzzy sets of the gas flow rate.](image-url)
In spite of this, it should be noted that the specification of GA parameters is not a strict issue and therefore does not have any sensitive influence on the results acquired. In principle, an appropriate population size can be regarded as a good balance between information diversity and convergence time, and the crossover rate should be quite large whereas the mutation rate be very small. The GA functions effectively as long as such basic requirements are satisfied.

4.1. Learning ability on training examples

To examine the learning ability of our method, all the available instances (293 data points) were used as training examples. First we set the coefficient $\beta$ in (5) equal to 0.3. Such a value assignment for $\beta$ caused the GA to find a four-rule model as shown below.

$R_1$: IF $[u(k) = \text{‘negative’} \text{ or ‘positive’}]$ and $[u(k - 1) = \text{‘positive’}]$ and $[u(k - 2) = \text{‘positive’}]$ and $[y(k - 1) = \text{‘middle’}]$ and $[y(k - 2) = \text{‘middle’}]$ and $[y(k - 3) = \text{‘middle’}]$, THEN $y^4(k) = 4.375 + 0.071u(k) + 0.099u(k - 1) - 0.437u(k - 2) + 1.806y(k - 1) - 1.151y(k - 2) + 0.263y(k - 3)$.

**Fuzzy set parameters:** $a_1 = -2.3865; a_2 = 1.398; b_1 = 41.20; b_2 = 55.47$.

Using the restricted sum as the $s$ norm and the algebraic product as the $t$ norm, the above fuzzy model has a mean square error (MSE) of 0.049 on the training data.

Next the value of $\beta$ was increased to 1.5, which implies that model complexity became a more important aspect in fitness evaluation compared with the former case when $\beta = 0.3$. The GA was guided in this case by the new cost function in searching for optimal rule conditions. As a result, premises of eight rules were identified as invalid, so that there are in fact only two rules in the rule base. The MSE performance of the second fuzzy model is 0.054 on the training data. Clearly, by defining a higher coefficient $\beta$, a little accuracy of the model was sacrificed in return for reduction in complexity.

Analysing the gas furnace data of Box and Jenkins is a well known benchmark problem on which many other modelling techniques have been studied. Table 1 compares the performances of our models with those from previous studies in terms of training errors, the number of input fuzzy sets as well as the number of fuzzy rules required. The first index here reflects the modelling accuracy on the training data, while the other two indices correspond to the model’s complexity and description capability.

**Remark 1:** The sizes of the rule bases generated in this paper (case 1 and case 2) were determined by the GA dynamically, whereas the rule numbers of the most other models in table 1 were fixed by human users prior to the procedure of model building.

**Remark 2:** The proposed method also enables different trade-offs between model accuracy and complexity. As stated above, two alternative fuzzy models were derived by defining different $\beta$ coefficients in (5). The values of $\beta$ reflects human demand or preference in model construction.

**Remark 3:** Our fuzzy model in case 1 has an MSE index of 0.049 on the training data, which is evidently much smaller than those of any other models listed in table 1. On the other hand, this model still contains a small number of rules in spite of its very high modelling accuracy.

**Remark 4:** The modelling accuracy of our model in case 2 is little superior to that of the model of Kim et al. (1997) but considerably better than the results of the

4.2. Generalization ability for test examples

To examine the generalization ability of our method on test instances, the available data set was divided into two groups with 147 examples as the training data and the other 146 examples as the test data. The learning algorithm was executed on the basis of the training data to build a fuzzy model, which was subsequently utilized to predict the outputs of the test data. As a consequence, a two rule model was extracted from the 147 training examples and it has an MSE of 0.073 on the test data. Figures 8 and 9 depict the outputs from our trained model and its predicting errors respectively on the test data. It can be observed from figure 9 that the predicting errors are usually less than 1.0, except on one point with the error of about 1.56. Although a comparison of the generalization ability of our model with those of others is not available owing to the lack of statements about test errors in the open literature, the MSE acquired here on the test data is quite satisfactory and even smaller than six training errors listed in table 1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of inputs</th>
<th>Number of input membership functions</th>
<th>Number of rules</th>
<th>MSE on the training data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tong (1980)</td>
<td>2</td>
<td>13</td>
<td>19</td>
<td>0.469</td>
</tr>
<tr>
<td>Pedrycz (1984)</td>
<td>2</td>
<td>18</td>
<td>81</td>
<td>0.320</td>
</tr>
<tr>
<td>Xu and Lu (1987)</td>
<td>2</td>
<td>10</td>
<td>25</td>
<td>0.328</td>
</tr>
<tr>
<td>Box and Jenkins (1970)</td>
<td>6</td>
<td>—</td>
<td>25</td>
<td>0.202</td>
</tr>
<tr>
<td>Sugeno and Yasukawa (1993)</td>
<td>3</td>
<td>18</td>
<td>6</td>
<td>0.190</td>
</tr>
<tr>
<td>Wang and Langari (1995)</td>
<td>6</td>
<td>12</td>
<td>2</td>
<td>0.066</td>
</tr>
<tr>
<td>Sugeno and Tanaka (1991)</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>0.068</td>
</tr>
<tr>
<td>Lin and Cunningham (1995)</td>
<td>5</td>
<td>20</td>
<td>4</td>
<td>0.071</td>
</tr>
<tr>
<td>Emami et al. (1998)</td>
<td>3</td>
<td>18</td>
<td>6</td>
<td>0.158</td>
</tr>
<tr>
<td>Kim et al. (1997)</td>
<td>6</td>
<td>12</td>
<td>2</td>
<td>0.055</td>
</tr>
<tr>
<td>Our model (case 1)</td>
<td>6</td>
<td>6</td>
<td>4</td>
<td>0.049</td>
</tr>
<tr>
<td>Our model (case 2)</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.054</td>
</tr>
</tbody>
</table>

Table 1. Performance comparison with other models
5. Conclusion
A key issue in constructing fuzzy models is to specify rule conditions, which determine the structure of the knowledge base. Taking into account all canonical AND connectives of input fuzzy sets as rule antecedents is not suitable in the cases of multiple input variables, since the number of rules generated increases exponentially with increasing input dimension. This paper proposes a new approach to designing fuzzy models by means of premise learning. Our method has the following important features.

(1) The rule premises are optimized via the GA rather than enumerated in terms of every canonical AND combination of input fuzzy sets.

(2) An economical rule base is made possible due to (a) selection of important regions of input domain to be covered and (b) general rule premises allowed in the coding scheme.

(3) The size of the rule base can be adjusted in the sense that we do not have to specify the exact rule number in advance. What is needed is only a ‘guess’ about how many rules appear sufficient to solve the problem under consideration.

(4) Excellent approximation accuracy can be achieved on both training and test data.

(5) The fuzzy model learned is transparent and easily understandable.

A major limitation of the current work lies in the determination of the upper limit of the rule number. If this upper limit is supposed to be too small, no satisfactory solution can be found. On the other hand, a too ‘generous’ estimation means a chromosome length much longer than necessary, so that the search space for the GA will become extremely large. Heuristic knowledge or experiences play a key role in making a favourable estimation about the sufficient amount of rules to achieve good accuracy. Further research attempts will be concentrated on automatic identification of the number of rules without a priori information.

Appendix A
For a chromosome (hybrid string) encoding $R_S$ valid rule conditions, we should derive $R_S$ consequent linear functions $y^j = d_0^j + d_1^j x_1 + \cdots + d_n^j x_n (j = 1, \ldots, R_S)$ to obtain a complete fuzzy model. Suppose that the training set consists of $N_T$ data points. The $k$th input vector and output are represented by $X(k) = (x_1(k), x_2(k), \ldots, x_n(k))$ and $y(k)$ respectively. Our task is to determine the coefficients in the linear functions by minimizing the model error on the example points in the training set. In principle, all the coefficients can be estimated by the least-square method simultaneously, if sufficient training data are available, that is $N_T \geq (n + 1) R_S$.

In practice, however, such a global parameter estimation is often connected with a heavy computational burden. Considering that conclusion identification is nested in the premise optimization for fitness evaluation of the GA, the process of estimating consequent parameters should be made as rapidly as possible. For this purpose rule conclusions are identified locally with the weighted least-square method in this paper.

Let

$$G_L = \begin{pmatrix} 1 & x_1(1) & x_2(1) & \cdots & x_n(1) \\ 1 & x_1(2) & x_2(2) & \cdots & x_n(2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1(N_T) & x_2(N_T) & \cdots & x_n(N_T) \end{pmatrix}$$

(A1)

and

$$Q_j = \begin{pmatrix} t_j(1) \\ t_j(2) \\ \vdots \\ t_j(N_T) \end{pmatrix}$$

(A2)

where $t_j(k)$ represents membership grade of the $k$th input vector with respect to the $j$th (valid) rule condition. Then consequent parameters corresponding to the $j$th rule condition are obtained by

$$(a_0^j \ a_1^j \ \cdots \ a_n^j)^T = (G_L^T Q_j G_L)^{-1} G_L^T Q_j Y.$$ 

(A3)

The main idea of the local estimation is that consequent parameters of an individual rule are determined independently without considering interactions of other rules. The inclusion of the matrix $Q_j$ in (A3) enables measuring data to be weighted according to their membership grades to the rule premise. Data points far from input region of the rule premise play almost no role in consequence determination, and vice versa.

Acknowledgements
The author would like to thank the anonymous referees for their very constructive comments and suggestions.

References
Evolutionary learning of rule premises for fuzzy modelling


