Fuzzy Rule Base Model Identification Techniques[1]

Kóczy, L.T. [*,] - Botzheim, J. [*]**

*Department of Telecommunication and Telematics, Budapest University of Technology and Economics, H-1117 Budapest, Magyar tudósok krt 2, Hungary

**Institute of Information Technology and Electrical Engineering, Széchenyi István University, Győr, Hungary

Abstract: *Fuzzy model identification techniques are presented in this paper. The paper discusses how fuzzy rules from a pattern set can be extracted without human interference. There are several methods used for rule extraction. Some of these were inspired by biological evolution. Other algorithms have been developed initially for neural networks and can be adapted to fuzzy systems. Fuzzy clustering has also been used for rule extraction. The fuzzy rule interpolation method and hierarchical rule bases are introduced. Combining fuzzy rule interpolation with the use of hierarchically structured fuzzy rule bases leads to the reduction of the fuzzy algorithms' complexity.*

1 Introduction

In the application of fuzzy systems to modeling and control one of the most important tasks is to find the optimal rule base. This might be given by a human expert or might be given a priori by the linguistic description of the modeled system. If, however, neither a suitable expert, nor the necessary linguistic descriptions are available, the system has to be designed by other methods based on numerical data. In training, the objective is to tune the membership functions in the fuzzy system so that the system performs a desired mapping of input to output. Besides the extraction of rules, it is important to decrease the computational complexity. If a fuzzy model contains k variables and maximum T linguistic (or other fuzzy) terms in each dimension, the number of necessary rules is of order $O(T^k)$. Decreasing T, or k, or both can decrease this expression. The first method leads to sparse rule bases and rule interpolation, first introduced by Kóczy and Hirota (see e.g. [1,2]). The second one, more effective, aims to reduce the dimension of the sub-rule bases (k's) by using meta-levels or hierarchical

[1] Research supported by the National Scientific Research Fund OTKA T034233 and T034212, a Széchenyi University Research Grant 2002 and the National Research and Development Project Grant 2/0015/2002

structures of fuzzy rule bases. The combination of the two was first attempted in [3].

The paper is organised as follows. Section 2 describes the bacterial algorithm. The Levenberg-Marquardt algorithm is shown in the Section 3. Section 4 and 5 introduces fuzzy clustering based rule extraction technique. Fuzzy rule interpolation and hierarchical rule bases are presented in Section 6, 7 and 8. Section 9 is the conclusion of the paper.

2 The Bacterial Algorithm

Nature inspired some evolutionary optimisation algorithms suitable for global optimisation of even non-linear, high-dimensional, multimodal, and discontinuous problems. The original genetic algorithm (GA) was developed by Holland [4] and was based on the process of evolution of biological organisms. A more recent approach is the bacterial evolutionary algorithm (BEA). The operations of bacterial algorithm were inspired by the microbial evolution phenomenon.

2.1 The Encoding Method

The membership functions are described by four parameters with the four breakpoints of the trapezium. Moreover the membership functions are identified by the two indices i and j. So, an input membership function A_{ij} belongs to the i^{th} rule and the j^{th} input variable and has four parameters: $a_{ij},b_{ij},c_{ij},d_{ij}$. An output membership function B_i belongs to the i^{th} rule and its parameters are: a_i,b_i,c_i,d_i.

The encoding method of a fuzzy system with two inputs and one output can be seen in [6]. An individual (bacterium) consists of the parameters of the corresponding fuzzy rule base.

2.2 The Algorithm

2.2.1 Generating The Initial Population

First the initial (random) bacteria population is created. The population consists of n chromosomes (bacteria). This means that all membership functions in the chromosomes must be randomly initialised. The initial number of rules in one chromosome is N_{max}. So, $n(k+1)N_{max}$ membership functions are created, where k is the number of input variables in the given problem.

2.2.2 Bacterial Mutation

The bacterial mutation is applied to each chromosome one by one [5]. First, $m - 1$ copies (clones) of the rule base are generated. Then a certain part of the chromosome [5] is randomly selected and the parameters of this selected part are randomly changed in each clone (mutation). Next all the clones and the original bacterium are evaluated by an error criterion. The best individual transfers the mutated part into the other individuals. This cycle is repeated for the remaining parts, until all parts of the chromosome have been mutated and tested. At the end the best rule base is kept and the remaining $m - 1$ are discharged.

2.2.3 Gene Transfer

The gene transfer operation allows the recombination of genetic information between two bacteria.

1. First the population must be devided into two halves. The better bacteria are called superior half, the other bacteria are called inferior half [5].

2. One bacterium is randomly chosen from the superior half, this will be the source bacterium, and another is randomly chosen from the inferior half, this will be the destination bacterium.

3. A "good" part from the source bacterium is chosen and this part can overwrite a not-so-good part of the destination bacterium or simple be added [5].

4. 1, 2, and 3 are repeated for $N_{\inf}$ times, where $N_{\inf}$ is the number of "infections" per generation [5].

2.2.4 Stop Condition

If the population satisfies a stop condition or the maximum generation number is reached then the algorithm ends, otherwise it returns to the bacterial mutation step.

3 The Levenberg-Marquardt Algorithm

There are other identification methods, which have been originally applied to train neural networks. One of the most successful neural networks training algorithm, the Levenberg-Marquardt algorithm [8] is discussed in this section. Optimal rule base means that the distances between the targets and the corresponding output vector gives smaller error than in the case of another rule base. The error is measured by the following function:

$$E = \frac{1}{2}\sum_{p=1}^{P}(t^{(p)} - y^{(p)})^2 \tag{1}$$

where P is the number of patterns in the pattern set, $t^{(p)}$ is the p^{th} target vector, $y^{(p)}$ is the p^{th} output vector. The most used method to minimise (1) is the Error-Back-Propagation (BP) algorithm, which is a steepest descent algorithm. A newer method is the Levenberg-Marquardt algorithm. Denoting the parameter vector by $\underline{z}$, and the Jacobean matrix by $\underline{\underline{J}}$:

$$\underline{\underline{J}}[k] = \frac{\partial y[\underline{z}[k]]}{\partial \underline{z}^T[k]} \tag{2}$$

$\underline{s}[k] = \underline{z}[k] - \underline{z}[k-1]$ the LM update, is given as the solution of

$$(\underline{\underline{J}}^T[k]\underline{\underline{J}}[k] + \alpha \underline{\underline{I}})\underline{s}[k] = -\underline{g}[k] = -\underline{\underline{J}}^T[k]\underline{e}[k] \tag{3}$$

In (3), α is a regularization parameter, which controls the both the search direction and the magnitude of the update. (3) can be recast as [9]:

$$\underline{s}[k] = -\begin{bmatrix} \underline{\underline{J}}[k] \\ \sqrt{\alpha}\,\underline{\underline{I}} \end{bmatrix}^{+} \begin{bmatrix} \underline{e}[k] \\ \underline{0} \end{bmatrix} \tag{4}$$

The complexity of this operation is of $O(n^3)$, where n is the number of columns of $\underline{\underline{J}}$.

If we apply this algorithm in fuzzy systems then the parameters z must be found. The structure of the fuzzy system is the following:

A grid in the input space is defined. Vectors of *knots* must be defined, one for each input dimension. These vectors determine the place of the membership functions. In each i^{th} axis $\lambda_{i,j}$ will be defined, where $j = 0,1,\ldots,r_i$. They are arranged in such a way that

$$\lambda_{i,0} \le \lambda_{i,1} \le \ldots \le \lambda_{i,r_i} \tag{5}$$

$\lambda_{i,0}$ is the minimal and λ_{i,r_i} is the maximal value of the i^{th} input.

Each output can be defined also by four parameters: $v_{r,1}, v_{r,2}, v_{r,3}, v_{r,4}$ in the r^{th} rule. The task is to find the location of each λ and ν parameter. This can be solved by the Jacobean computation which is described in [9] and for fuzzy systems in [7].

4 Clustering-Based Rule Extraction Technique

Recently, clustering-based approaches have been proposed for rule extraction [10, 11]. Most of the techniques use the idea of partitioning the input space into fixed regions to form the antecedents of the fuzzy rules. Although these techniques have the advantage of efficiency, they may lead to the creation of a dense rule-base that suffers from rule explosion. In general, the number of rules generated is t^d where *d* is the number of input dimensions and *t* is the terms per input. In this case, the number of rules generated grows exponentially with the increase of input dimensions. Due to this reason, the techniques are not suited for generating fuzzy rule-bases that have a large number of input dimensions.

Among the rule extraction techniques proposed in the literature, Sugeno and Yasukawa's [12] technique (abbreviated as SY method hereafter) is one of the earliest works that emphasize the generation of a sparse rule-base. The SY approach clusters only the output data and induces the rules by computing the projections to the input domains of the cylindrical extensions of the fuzzy clusters. This way, the method produces only the necessary number of rules for the input-output sample data (more details later). The paper [12] discusses the proposed technique at the methodological level leaving out some implementation details. The SY technique was further examined in [16] where additional readily implementable techniques are propose to complete the modeling methodology.

In the first step of SY modelling, the Regularity criterion [13] is used to assist in the identification of 'true' input variables that have significant influence on the output. The input variables that have less or no influence on the output are ignored for the rest of the process. The true input variables are then used in the actual rule extraction process. The rule extraction process starts with the determinition of the partition of the output space. This is done by using fuzzy c-means clustering [14] (see section 5).

For each output fuzzy cluster B_i resulting from the fuzzy c-means clustering, a cluster in the input space A_i can be induced. The input cluster can be projected onto the various input dimensions to produce rules of the form:

If x_1 is A_{i1} and x_2 is A_{i2} and … x_n is A_{in} then y is B_i

5 Fuzzy C-Means Clustering

Given a set of data, Fuzzy c-Means clustering (FCMC) performs clustering by iteratively searching for a set of fuzzy partitions and the associated cluster centers that represent the structure of the data as best as possible. The FCMC algorithm relies on the user to specify the number of clusters present in the set of data to be clustered. Given the number of cluster *c*, FCMC partitions the data X = $\{x_1,x_2,\ldots,x_n\}$ into *c* fuzzy partitions by minimizing the within group sum of squared error objective function as follows (6).

$$J_m(U,V) = \sum_{k=1}^{n}\sum_{i=1}^{c}(U_{ik})^m \, \| x_k - v_i \|^2, \; 1 \le m \le \infty, \tag{6}$$

where J_m(U,V) is the sum of squared error for the set of fuzzy clusters represented by the membership matrix U, and the associated set of cluster centers V. ||.|| is some inner product-induced norm. In the formula, $\|x_k - v_i\|^2$ represents the distance between the data x_k and the cluster center v_i. The squared error is used as a performance index that measures the weighted sum of distances between cluster centers and elements in the corresponding fuzzy clusters. The number *m* governs the influence of membership grades in the performance index. The partition becomes fuzzier with increasing *m* and it is proven that the FCMC algorithm converges for any $m \in (1,\infty)$. The necessary conditions for (6) to reach its minimum are

$$U_{ik} = \left(\sum_{j=1}^{c} \left(\frac{\| x_k - v_i \|}{\| x_k - v_j \|} \right)^{2/(m-1)} \right)^{-1} \qquad \forall i, \forall k, \tag{7}$$

and

$$v_i = \frac{\sum_{k=1}^{n} (U_{ik})^m x_k}{\sum_{k=1}^{n} (U_{ik})^m}, \tag{8}$$

In each iteration of the FCMC algorithm, matrix U is computed using (7) and the associated cluster centers are computed as (8). This is followed by computing the square error in (6). The algorithm stops when either the error is below a certain tolerance value or its improvement over the previous iteration is below a certain threshold.

The FCMC algorithm cannot be used in situations where the number of clusters in a set of data is not known in advance. Since the introduction of FCMC, a reasonable amount of work has been done on finding the optimal number of cluster in a set of data. This is referred to as the cluster validity problem. The optimal number of clusters are determined by means of a criterion, known as the cluster validity index. Sugeno and Fukuyama proposed the following cluster validitiy index in [15].

$$S(c) = \sum_{k=1}^{n}\sum_{i=1}^{c}(U_{ik})^m (\| x_k - v_i \|^2 - \| v_i - \bar{x} \|^2) \quad 2 < c < n, \tag{9}$$

where n is the number of data points to be clustered; c is the number of clusters; x_k is the k^{th} data, $\bar{x}$ is the average of data; v_i is the i^{th} cluster center; U_{ik} is the membership degree of the k^{th} data with respect to the i^{th} cluster and m is the fuzzy exponent. The number of clusters, c, is determined so that $S(c)$ reaches a local minimum as c increases. The terms $\| x_k - v_i \|$ and $\| vi- \bar{x} \|$ represent the variance in each cluster and variance between clusters respectively. Therefore, the optimal number of clusters is found by minimizing the distance between data to the corresponding cluster center and maximizes the distance between data in different clusters. Other cluster validity indexes can be found in [17].

6 Fuzzy Rule Interpolation

In the following sections two complexity reduction techniques will be presented: the fuzzy rule interpolation and the use of hierarchical rule bases. In these sections we shall use the vector representation of fuzzy sets, which assigns a vector of its characteristic points to every fuzzy set. The representation of the piecewise linear fuzzy set A will be denoted by vector $\underline{a} = [a_{-m} ,...,a_0 ,...,a_n]$, where a_k ($k \in [-m,n]$) are the characteristic points of A and a_0 is the reference point of A having membership degree one. A partial ordering among CNF fuzzy sets (convex and normal fuzzy sets) is defined as: $A \prec B$ if $a_k \le b_k$ ($k \in [-m,n]$).

The basic idea of fuzzy rule interpolation (KH-interpolation) is formulated in the *Fundamental Equation of Rule Interpolation* (FERI): $D(A^*, A_1) : D(A^*, A_2) = D(B^*, B_1) : D(B^*, B_2)$.

In this equation A^* and B^* denote the observation and the corresponding conclusion, while $R_1 = A_1 \to B_2, R_2 = A_2 \to B_2$ are the rules to be interpolated, such that $A_1 \prec A^* \prec A_2$ and $B_1 \prec B_2$. If in some sense D denotes the Euclidean distance between two symbols, the solution for B^* results in simple linear interpolation. If

$D = \tilde{d}$ (the fuzzy distance family), linear interpolation between corresponding α-cuts is performed and the generated conclusion can be computed as below, (as it is first described in [3]):

$$b_k^* = \frac{\dfrac{b_{1k}}{d(a_{1k}, a_k^*)} + \dfrac{b_{2k}}{d(a_{2k}, a_k^*)}}{\dfrac{1}{d(a_{1k}, a_k^*)} + \dfrac{1}{d(a_{2k}, a_k^*)}} \tag{10}$$

where the first index (*1* or *2*) represents the number of the rule, while the second - *k* - the corresponding α-cut. From now on we shall consider $d(x, y) = |x - y|$, so that (10) becomes: ${}^{KH}b_k^* = (1 - \lambda_k) b_{1k} + \lambda_k b_{2k}$, where $\lambda_k = (a_k^* - a_{1k}) / (a_{2k} - a_{1k})$ (for the left and right side respectively).

7 Hierarchical Fuzzy Rule Bases

The input space $X = X_1 \times X_2 \times \ldots \times X_m$ can be decomposed, so that some of its components, e.g. $Z_0 = X_1 \times X_2 \times \ldots \times X_p$ determine a subspace of *X* (*p* < *m*), so that in Z_0 a partition $\Pi = \{D_1, D_2, \ldots, D_n\}$ can be determined: $\bigcup_{i=1}^{n} D_i = Z_0$.

In each element of Π, i.e. D_i, a sub-rule base R_i can be constructed with local validity. In the worst case, each sub-rule base refers to exactly $X / Z_0 = X_{p+1} \times \ldots \times X_m$. The complexity of the whole rule base $O(T^m)$ is not decreased, as the size of R_0 is $O(T^p)$, and each R_i, *i* > *0,* is of order $O(T^{m-p})$, $O(T^p) \times O(T^{m-p}) = O(T^m)$.

A way to decrease the complexity would be finding in each D_i a proper subset of $\{X_{p+1} \times \ldots \times X_m\}$, so that each R_i contains only less than *m-p* input variables. The task is of finding such a partition is often difficult, if not impossible, (sometimes such partition does not even exist).

There are cases when, locally, some variables unambiguously dominate the behaviour of the system, and consequently the omission of the other variables allows an acceptably accurate approximation. The bordering regions of the local domains might not be crisp or even worse, these domains overlap. For example, there is a region D_1, where the proper subspace Z_1 dominates, and another region D_2, where another proper subspace Z_2 is sufficient for the description of the system, however, in the region between D_1 and D_2 all variables in $[Z_1 \times Z_2]$ play a significant role ($[\cdot \times \cdot]$ denoting the space that contains all variable that occur in either argument within the brackets). In this case, sparse fuzzy partitions can be used, so that in each element of the partition a proper subset of the remaining input state variables is identified as exclusively dominant. Such a sparse fuzzy partition can be described as follows: $\hat{\Pi} = \{D_1, D_2, ..., D_n\}$ and $\bigcup_{i=1}^{n} Core(D_i) \subset Z_0$ in the proper sense (fuzzy partition). Even $\bigcup_{i=1}^{n} Supp(D_i) \subset Z_0$ is possible (sparse partition). If the fuzzy partition chosen is informative enough concerning the behaviour of the system, it is possible to interpolate its model among the elements of $\hat{\Pi}$, as we shall see below.

Each element D_i will determine a sub-rule base R_i referring to another subset of variables. The technical difficulty is how to combine the ”sub-conclusions” B_i^* with the help of R_0 into the final conclusion.

E.g., let us assume that the fuzzy partition has only two elements: $\hat{\Pi} = \{D_1, D_2\}$, and that $[Z_1 \times Z_2] = Z_1 \times Z_2$,i.e., Z_1 and Z_2 have no common component X_i .Consequently, $X = Z_0 \times Z_1 \times Z_2$. The rule base will have the following structure:

R_0: If z_0 is D_1 then use R_1

If z_0 is D_2 then use R_2

R_1:	If z_1 is A_{11} then y is B_{11}	R_2:	If z_2 is A_{21} then y is B_{21}
	If z_1 is A_{12} then y is B_{12}		If z_2 is A_{22} then y is B_{22}
	…		…
	If z_1 is A_{1r_1} then y is B_{1r_1}		If z_2 is A_{2r_2} then y is B_{2r_2}

8 Hierarchical Rule Bases And Fuzzy Interpolation

Let us assume that the observation on X is A^* and its projections are: $A_0^* = A^* / Z_0$, $A_1^* = A^* / Z_1$, $A_2^* = A^* / Z_2$. Using the Fundamental Equation, the two sub-conclusions, obtained from the two sub-rule bases R_1 and R_2 are:

${}^{KH}b_k^{1*} = (1-\lambda_k^1)b_{1k}^1 + \lambda_k^1 b_{2k}^1$, and ${}^{KH}b_k^{2*} = (1-\lambda_k^2)b_{1k}^2 + \lambda_k^2 b_{2k}^2$ respectively.

(The superscript shows the reference to the rule base R_1 and R_2.)

Finally, by substituting the sub-conclusions into the meta-rule base we get:

$$ {}^{KH}b_k^* = (1-\lambda_k^0)b_k^{1*} + \lambda_k^0 b_k^{2*} \qquad (11) $$

The steps of the algorithm are the following :

1. Determine the projection A_0^* of the observation A^* to the subspace of the fuzzy partition $\hat{\Pi}$. Find the interpolating rules.

2. Determine λ_k^0.

3. For each R_i determine A_i^* the projection of A^* to Z_i. Find the interpolating rules in each R_i.

4. Determine the sub-conclusions for each sub-rule base R_i.

5. Using the sub-conclusions from step 4, compute the final conclusion according to (11).

9 Conclusion

Fuzzy model identification methods were described in this paper. Each algorithm has advantages and disadvantages too. The Levenberg-Marquardt algorithm often finds only the local minimum in the optimisation process. The bacterial algorithm can avoid the local minima, but this method gives just a quasi-optimal result. The clustering based rule extraction techniques have the advantage of being computationally efficient. The interpolation in hierarchical rule bases reduces the computational complexity. One of the drawbacks of this method is that it often results in abnormal conclusions, so the hierarchical structures are impossible to use.

Acknowledgement

This paper is partly based on research done in collaboration with: Prof. Tom D. Gedeon and Mr. Alex Chong (Murdoch University, Perth, Australia); Prof. Antonio Ruano (University of Algarve, Faro, Portugal); and Ms. Leila Muresan (Budapest University of Technology and Economics, Budapest, Hungary)

References

[1] Kóczy,L.T., Hirota, K. : Approximate reasoning by linear rule interpolation and general approximation, Intn'l J. of Approximate Reasoning, 9, 1993, pp. 197-225

[2] Kóczy,L.T., Hirota, K. : Interpolative reasoning with insufficient evidence in sparse fuzzy rule bases, Information Sciences 71 1993, pp. 169-201

[3] Kóczy,L.T., Hirota, K. : Interpolation in structured fuzzy rule bases, FUZZ-IEEE'93, San Francisco 1993, pp. 803-808

[4] Holland, J.H.: Adaptation in Nature and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence, MIT Press, Cambridge, 1992.

[5] Nawa, N.E. and Furuhashi, T.: Fuzzy System Parameters Discovery by Bacterial Evolutionary Algorithm, IEEE Tr. Fuzzy Systems **7** 1999, pp. 608-616.

[6] Botzheim, J., Hámori, B., and Kóczy, L.T.: Extracting trapezoidal membership functions of a fuzzy rule system by bacterial algorithm, 7th Fuzzy Days, Dortmund 2001,Springer-Verlag, pp. 218-227.

[7] Botzheim, J., Kóczy, L.T., Ruano, A.E.: Extension of the Levenberg-Marquardt algorithm for the extraction of trapezoidal and general piecewise linear fuzzy rules, 2002 World Congress on Computational Intelligence, Hawaii, USA

[8] Marquardt, D., An Algorithm for Least-Squares Estimation of Nonlinear Parameters, SIAM J. Appl. Math., 11, 1963, pp. 431-441

[9] Ruano, A.E., Cabrita, C., Oliveira, J.V., Kóczy, L.T., Tikk, D.: Supervised Training Algorithms for B-Spline Neural Networks and Fuzzy Systems, Joint 9th

IFSA World Congress and 20th NAFIPS International Conference, Vancouver, Canada, 2001.

[10] Wong, K.W., Fung, C.C., and Wong, P.M. A self-generating fuzzy rules inference systems for petrophysical properties prediction. in Proceedings of IEEE International Conference on Intelligent Processing Systems. 1997. Beijing.

[11] Wang, L.X. and Mendel, J.M., Generating fuzzy rules by learning from examples. IEEE Transactions on Systems, Man, and Cybernetics, 1992. **22**(6): p. 1414-1427.

[12] Sugeno, M. and Yasukawa, T., A fuzzy-logic-based approach to qualitative modeling. IEEE Transactions on Fuzzy Systems, 1993. **1**(1): p. 7-31.

[13] Ihara, J., Group method of data handling towards a modelling of complex systms - IV. Systems and Control (in Japanese), 1980. **24**: p. 158-168.

[14] Bezdek, J.C., Pattern Reconition with Fuzzy Objective Function Algorithms. 1981, New York: Plenum Press.

[15] Fukuyama, Y. and Sugeno, M. A new method of choosing the number of clusters for fuzzy c-means method. in Proceedings of the 5th Fuzzy System Symposium. 1989.

[16] Tikk, D., Biró, Gy., Gedeon, T.D., Kóczy, L.T., Yang, J.D., Improvements and Critique on Sugeno's and Yasukawa's Qualitative Modeling, IEEE Tr.on Fuzzy Systems, Vol.10. No.5., October 2002

[17] Yang, M.S., Wu, K.L., A New Validity Index For Fuzzy Clustering, in Proceedings of IEEE International Conference on Fuzzy Systems, December, Melbourne, 4 pages.