Johanyák, Z. C.: Sparse Fuzzy Model Identification Matlab Toolbox - RuleMaker Toolbox, IEEE 6th International Conference on Computational Cybernetics, November 27-29, 2008, Stara Lesná, Slovakia, pp. 69-74.

Sparse Fuzzy Model Identification Matlab Toolox – RuleMaker Toolbox

Z.C. Johanyák
Kecskemét College/Institute of Information Technologies, Kecskemét, Hungary
johanyak.csaba@gamf.kefo.hu

***Abstract*—Fuzzy systems applying a sparse rule base and a fuzzy rule interpolation based reasoning method are popular solutions in cases with partial knowledge of the modeled area or cases when the full coverage of the input space by rule antecedents would require too many rules.**

In several practical applications there is no human expert based knowledge; the fuzzy model has to be identified from sample data. This paper presents a freely available Matlab toolbox called RuleMaker that supports the automatic generation of a fuzzy model with low complexity. The implemented model identification methods are also reviewed.

I. Introduction

Fuzzy systems applying sparse rule bases are widely used when either one does not possesses the necessary knowledge for the full coverage of the input space or the complexity of the resulting system would be too high owing to the increased number of rules.

Due to the increasing popularity of the fuzzy rule interpolation (FRI) based reasoning methods (e.g. KH [20], IMUL [28], FIVE [22], GM [1], IRG [5], and LESFRI [14]) starting from the early 90s several methods aiming the automatic generation of sparse fuzzy systems have been developed. This paper reports the development of a Matlab toolbox called RuleMaker that is freely available under GNU GPL [12] and implements the fuzzy model identification methods RBE-DSS [15], RBE-SI [15] and ACP [13].

The rest of this paper is organized as follows. Section II gives a brief survey on the main tendencies in sparse fuzzy model identification and introduces shortly the implemented techniques. Section III presents the RuleMaker toolbox.

II. Sparse Fuzzy Rule Base Generation

A. Sparse Rule Base

The rule base of a fuzzy system is considered as sparse when there is at least one point of the input space for which there is no rule with an activation degree greater than a prescribed ε_0 value. In mathematical terms the degree of coverage (c) of the input space is described by

$$c = \arg\max_{\varepsilon}\Big(min_{i=1}^{N}\Big\{ max_{j=1}^{n_i}\Big\{ t\big(A_{i,j}, A_i^*\big)\Big\}\Big\} \geq \varepsilon, \quad \forall A_i^* \subset X_i \Big), \varepsilon \in [0, 1], \tag{1}$$

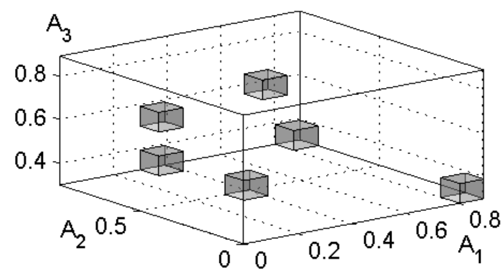

Figure 1. Antecedent space of a sparse fuzzy rule base

where X_i is the ith dimension of the antecedent space, A_i^* is the fuzzy set describing the observation in the ith antecedent dimension, A_{ij} is the jth linguistic term of the ith antecedent dimension, t is an arbitrary t-norm, n_i is the number of the linguistic terms of the ith antecedent dimension, N is the number of the antecedent dimensions, and $\arg\max_{\varepsilon}(.)$ calculates that ε value for which the expression in the parentheses takes its maximum. If $c > \varepsilon_0$ the rule base is called ε_0 covering (dense) otherwise it is considered as sparse.

Fig. 1 illustrates the antecedent space of a sparse rule base. The input of the system is three dimensional. Each rule is represented by a cuboid defined by the supports of the fuzzy sets referred in the antecedent part of the rule. The big cuboid containing the small ones defines the multidimensional input universe of discourse corresponding to the allowed input ranges of the system in each antecedent dimension.

B. Automatic Sparse Fuzzy Model Identification

In case of automatic fuzzy model identification one possesses a training data set that describes the expected behavior of the system in several points, i.e. contains a list of mating input-output values. The model identification algorithm seeks for those values of the system parameters for which the difference between the output values of the training data set and the output provided by the current fuzzy system in case of the same input is small. In most of the cases especially when the training data set contains a large amount of data an exact match cannot be reached. Therefore the fuzzy system gives only a more or less good approximation of the modeled input-output relation described by the sample data.

Usually the approximation can be improved by increasing the number of linguistic terms and rules. The situation is similar to the neural networks based modeling. However, the advantage of the application of a fuzzy system is given by its self explaining capability, i.e. one can extract from the tuned system IF-THEN rules that are easily interpretable and understandable by humans.

Sparse fuzzy systems with low complexity can be obtained following two different approaches. The first one starts with a dense rule base ($c > \varepsilon_0$ in (1)) [21], i.e. as step 0 a covering rule base should be generated and then the rules considered as non relevant ones are eliminated. The methods proposed by Botzheim, Cabrita, Kóczy and Ruano [3], Botzheim, Hámori and Kóczy [4], as well as by Kóczy, Botzheim and Gedeon [19] follow this way.

The second approach tries to create a sparse rule base directly from the available data. These methods can be categorized into three groups.

1. Methods trying to identify the so called optimal fuzzy rules (Kosko and Dickerson [23], Kosko [24]).
2. Methods based on fuzzy clustering (e.g. Chiu [6], Chong [9], Klawonn and Kruse [18], Sugeno and Yakusawa [25], Tikk, Gedeon, Kóczy and Bíró [26], Wong, Kóczy, Gedeon, Chong and Tikk [29] as well as Johanyák [13]).
3. Methods following the concept of rule base extension (e.g. Johanyák and Kovács [15]).

The current version of the RuleMaker Matlab ToolBox contains the implementation of the clustering based ACP method [13] and the implementation of the techniques RBE-DSS and RBE-SI [15] that follow the concept of rule base extension. All three methods consist of two steps. First a raw (starting) fuzzy system is generated followed by an iterative parameter identification process (second step).

C. *Automatic Fuzzy System Generation based on Fuzzy Clustering and Projection (ACP)*

The ACP method [13] was developed using some elements and concepts of the methods [7], [9], [25], [26], and [29]. The method consists of two main steps. First a starting (raw) fuzzy system is generated using clustering and projection. Next the quazi-optimal parameters of the fuzzy system are identified by an iterative process based on a hill climbing approach. Further on in course of the method description we suppose a multiple input single output (MISO) fuzzy system. If the modeled phenomenon is a multiple output (MIMO) one a separate model is generated for each output dimension.

The first step of the ACP method starts with the determination of the optimal cluster number for a fuzzy c-mean (FCM) type clustering of the output sample data. It is calculated by the help of a cluster validity index (CVI). One can find several CVIs in the literature. A survey and comparison of the available techniques was made by Wang and Zhang in [27]. ACP adapted the hybrid approach proposed by Chong, Gedeon and Kóczy [8] enhanced by the application of an upper limit for the cluster number. It was necessary because in case of some data sets the method resulted in a very high optimal cluster number, which was inacceptable for practical applications.

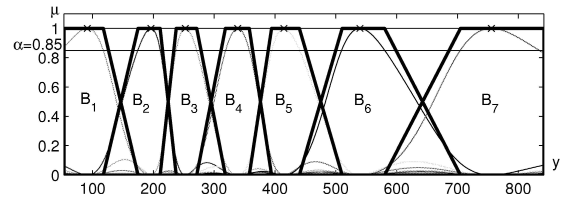

Figure 2. Creating a Ruspini partition from output clusters using a core estimation with an α=0.85 cutting level

The hybrid technique first calculates the Fukuyama-Sugeno [11] index followed by the examination of the mergeability of the adjacent clusters using the cluster merging index proposed by Chong, Gedeon and Kóczy [8]. After the determination of the optimal output cluster number one does a one-dimensional FCM clustering on the output data. The applied technique is based on the FCM proposed by Bezdek [2] extended with several modifications proposed in [13] in order to meet some special requirements of the automatic fuzzy model identification.

The output partition and its fuzzy sets are estimated from the previously identified clusters. ACP uses the most simple and straightforward technique for this task, which was originally proposed by Chong in [7]. This approach produces a Ruspini partition with trapezoidal shaped fuzzy sets by cutting the clusters at an α-level (usually α=0.85) and creating the set cores from the width values of the cuts. Owing to the Ruspini character of the partition one determines the supports of the sets automatically from the corresponding endpoints of the adjacent linguistic terms. Fig. 2. illustrates the creation of the consequent linguistic terms (bold lines) from the output clusters (thin lines).

Next the consequent linguistic terms are projected into the antecedent space. The projection means that for each output fuzzy set one selects the data rows whose output falls into the support of the set. Thus some data corresponding to the overlapping parts of the linguistic terms will be taken into consideration twice. Afterwards one does a one dimensional fuzzy clustering in each antecedent dimension on the current subset of the sample data. The cluster centers will be used later as reference points of the estimated antecedent fuzzy sets.

An identifier with three indices ($A_{i,j,k}$) is assigned to each cluster, where *i* is the ordinal number of the antecedent dimension, *j* is the ordinal number of the consequent fuzzy set in its partition, and *k* is the ordinal number of the cluster in the current clustering.

One can generate two or more rules for each output fuzzy set in the form

$$R_m : a_1 = A_{1,j,k_1}\ AND\ ...\ AND\ a_N = A_{N,j,k_N}\ ...\\ THEN\ b = B_j, \tag{2}$$

where *m* is the ordinal number of the rule, *N* is the number of antecedent dimensions. Thus one obtains several groups of cluster centers in each input dimension. Next they are interleaved and renumbered in order to get one cluster partition in each dimension. Some of the cluster centers will be identical or nearly identical. They are united applying a prescribed distance threshold whose value is a parameter of the method. After a cluster union the related rules are also corrected or united.

One estimates the antecedent partitions and linguistic terms using the same methods as in the case of the consequent partition. The resulting fuzzy model is further improved in the second step of ACP by a parameter identification (tuning) process. The algorithm uses an iterative hill climbing approach.

After each step (each parameter modification) the resulting fuzzy system is evaluated by the means of a performance index, which gives information about the difference between the prescribed output given by the sample data set and the calculated data set using the fuzzy system.

The toolbox contains seven options for the selection of the performance index. Probably the most used and straightforward one is the relative value of the mean square error expressed in percentage of the output range (RMSEP)

$$PI_{RMSEP} = \sqrt{\frac{\sum_{j=1}^{M}\left(y_j - \hat{y}_j\right)^2}{M}} \cdot DR_{N+1}^{-1} \cdot 100\,, \tag{3}$$

where M is the number of data points in the sample, y_j is the *j*th prescribed output value, $\hat{y}_j$ is the *j*th calculated output value, and DR_{N+1} is the range of the output.

The consequents in (2) can be constant or affine functions of the inputs. Applications can be found in fuzzy control systems [18], [30].

The algorithm investigates the parameters one-by-one in course of each iteration cycle. First the system is evaluated (PI_{RMSEP}) with the current parameter set. Next one calculates two new values for the actual parameter one by decreasing the original value and one by increasing the original value. The step (st_i) is calculated in function of the range (DR_i) in the actual (*i*) dimension

$$st_i = C \cdot DR_i\,, \tag{4}$$

where *C* is a constant that applies to each dimension. After the evaluation of the system with he two new parameter values one keeps that value from the available three ones (original and two new values), which results the best system performance. An iteration cycle contains the investigation of all parameters once.

At the end of the cycle one compares the actual performance (PI_k) with the performance of the system measured at the end of the previous cycle (PI_{k-1}). If the amelioration of *PI* is greater than a threshold the value of *C* is doubled otherwise *C* is decreased to its half. When it becomes smaller than a threshold ($C_{\min}$) for the first time the process receives a second chance by increasing *C* to its original value. The algorithm stops when either *C* reaches its minimum for the second time or the prescribed number of iteration cycles is reached or the performance index of the system becomes better than a threshold value.

D. Automatic Fuzzy Model Identification by Rule Base Extension

The concept of rule base extension (RBE) [15] starts the fuzzy model identification with the creation of two rules, one describing the minimum output and one describing the maximum output, i.e. the reference points of the consequent sets will be identical with the mentioned extreme points. If the minimum (maximum) output is reached in case of several different input values one chooses those values as reference points for the antecedent sets that are closer to the lower or upper bounds of the corresponding input domain. In case of trapezoidal shaped membership functions one determines a default core and support width for each dimension depending on the range of the domain.

Similar to ACP the RBE based methods (RBE-DSS and RBE-SI) [15] also have a second step aiming the improvement of the original fuzzy system in order to reach a quasi-optimal performance index. The algorithm is similar to the one applied in the second step of ACP. It differs mainly in the handling of the local optimum of *PI*. In case of RBE when the coefficient (C) of the step (st_i) reaches its minimum one generates a new rule fitting the output value where the difference between the sample output and the calculated output is maximal.

Rule Base Extension using Default Set Shapes (RBE-DSS) creates the fuzzy sets of the new rule with the same core and support width values (supposing trapezoidal shaped membership functions) as the ones applied for the first two rules. However, the insertion of the new rule sometimes results in a temporary deterioration of the performance index.

In order to avoid this problem Rule Base Extension using Set Interpolation (RBE-SI) calculates the shape of the antecedent and consequent linguistic terms based on set interpolation. It is not mandatory but it could be expedient to choose the set interpolation method conform to the applied fuzzy rule interpolation based reasoning technique.

III. RuleMaker Toolbox

A. General description

The RuleMaker toolbox is implemented in Matlab and is available for download under GNU General Public License [12] from the website [10]. It was developed and tested using Matlab 2006b. RuleMaker aims the generation of a low complexity fuzzy system usually with a sparse rule base. Its current version contains the implementation of the methods ACP, RBE-DSS and RBE-SI. The toolbox is a collection of Matlab functions that can be called from command line and from other Matlab programs. The package also contains software with graphical user interface ensuring an easy-to-use access to

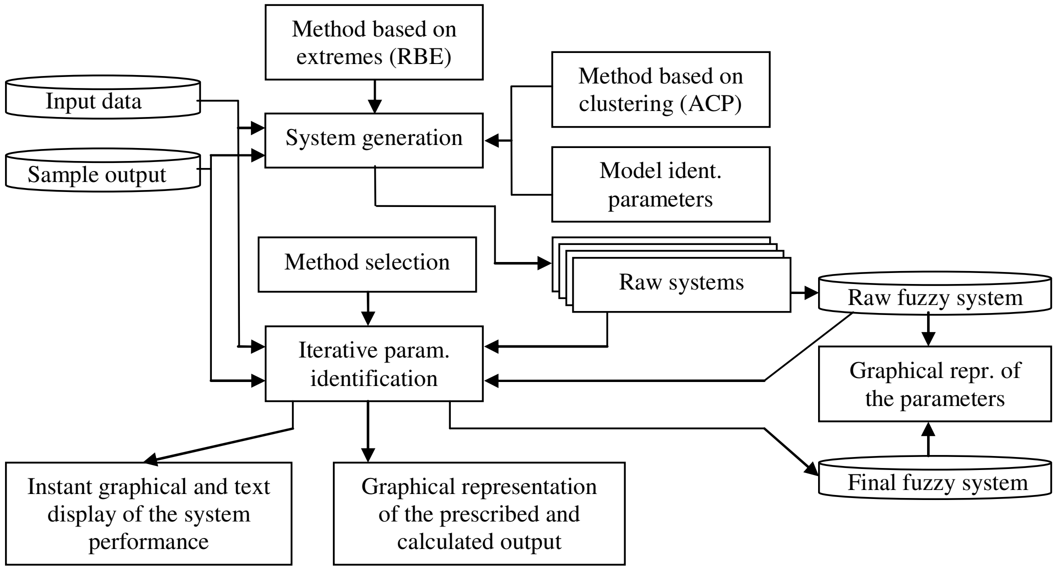

Figure 3. Internal structure of the program

all the services offered by the toolbox. Its internal structure is presented in fig. 3.

B. *Usage of the GUI*

The graphical user interface of the toolbox can be started by typing in the Matlab Command Window the command *RuleMaker*. Conform to the implemented fuzzy model identification methods one can distinguish two main areas in the functionality of the GUI program. The first one deals with the generation of the raw fuzzy system from the sample data. The training data should be loaded in two separate text files, one containing the output values and one containing the input values. In both cases each row corresponds to one data point. If the input or output is multi-dimensional the co-ordinates in the different dimensions are separated by tabulators.

One can generate the raw fuzzy system either by clustering or by the extreme values based approach. The system can be saved in a text file using the same format as the Fuzzy Rule Interpolation Matlab toolbox [17]. The program generates a separate raw system for each output dimension in case of sample data with multi-dimensional output.

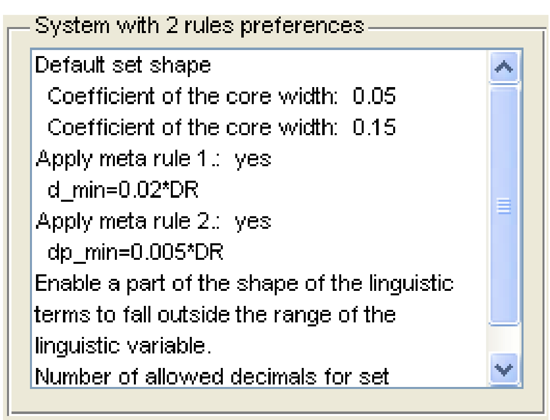

Figure 4. Parameters of the raw fuzzy system generation

The current version (1.0.0) of the toolbox supports only the trapezoidal shaped membership functions, which includes with corresponding parameterization the triangle and the singleton shaped fuzzy sets as well. The default core and support width values and some other parameters can be set by the user through dialog boxes. They are also displayed in a textbox in the main window (see fig. 4).

The program makes possible the graphical representation of the input and output partitions (see e.g. fig. 5) separately, as well as a three dimensional representation of the rule base or antecedent space if the number of antecedent dimensions does not exceed three. If the fuzzy model has only one input dimension the program offers a three dimensional representation of the rule base each rule being indicated by a pyramid defined by the antecedent and consequent membership functions (see e.g. fig. 6). If the number of input dimensions is two one can use the pyramid-type representation for the visualization of the antecedent space. Here the pyramids are defined by the antecedent fuzzy sets. However, the user can choose the cuboid-type representation (e.g. fig. 1) where the rule consequents still can be drawn into the visualization. In this case the edges of a cuboid represent the supports of the antecedent and consequent linguistic

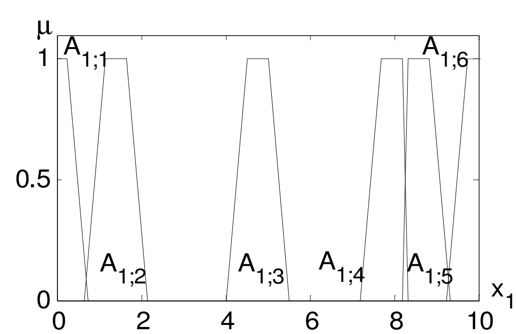

Figure 5. Antecedent partition of a fuzzy system

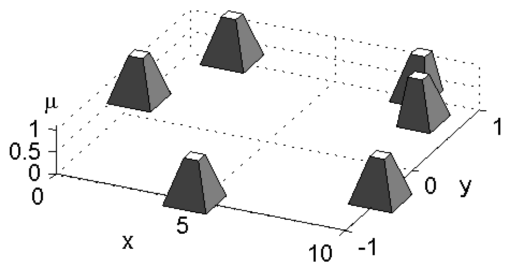

Figure 6. 3D representation of a SISO fuzzy system. Each pyramid represents a rule and is defined by the antecedent and consequent membership functions

terms of a rule. The drawback of this type is that other parts of the sets' shape cannot be visualized. In case of fuzzy systems having three input dimensions only the cuboid-type representation is available. It can be used for the visualization of the rule antecedents.

The second main area of the program functionality deals with the parameter identification. After each step (modification of a parameter) the system is evaluated using a performance index. The user can choose from seven PIs. They are the mean square of the error (MSE), the relative value of the MSE compared to the range of the output variable (RMSE), the RMSEP (3), the correlation factor between the prescribed output and the calculated one (R), the transformed value of R (RR) [13], the mean of the differences in absolute value between the prescribed output and the calculated one (AD), as well as the relative value of AD compared to the range of the output variable (ADP).

The fuzzy inference is done by the help of the Fuzzy Rule Interpolation (FRI) Matlab toolbox [17]. Ten fuzzy inference methods can be selected. The RuleMaker toolbox offers four fuzzy set parameterization types (breakpoints, relative distances, reference points, and a core endpoints based one that always ensures the Ruspini character of the partition). Beside the determination of the performance index the calculated and prescribed output values can be compared in a diagram. Fig. 7 illustrates the graphical representation of the momentary output. The abscissa indicates the ordinal number of the data points. The output from the sample data is represented by (blue) circles and the output calculated by the fuzzy system for same input is represented by (red) pentagrams.

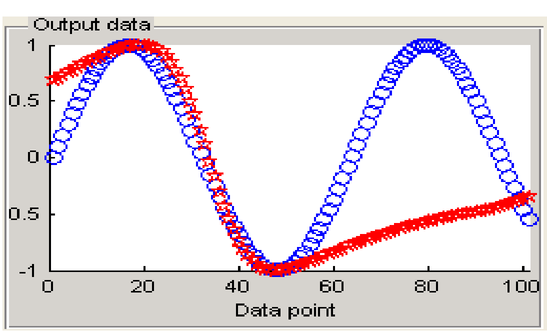

Figure 7. Prescribed (circles) and calculated (pentagrams) output in function of the ordinal number of the data points

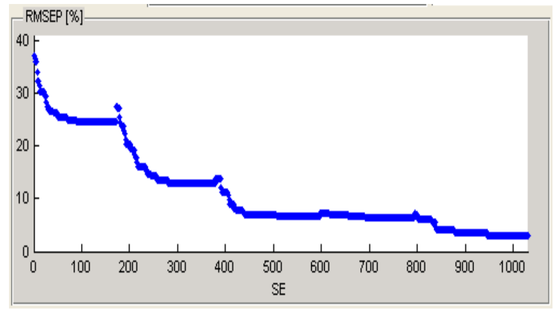

Figure 8. Variation of the performance index (RMSEP) in course of the parameter identification represented in function of the number of system evaluations (SE)

One can also get a comprehensive view of the tuning from a diagram (see fig. 8) showing the variation of the performance index (RMSEP on fig. 8). Here the abscissa shows the number of system evaluations (SE). The fuzzy model can be saved after each iteration step. The parameter identification usually is a time and computational resources consuming task. Therefore the user can pause or stop the process at any time by the help of push buttons.

C. Further development plans

The RuleMaker Toolbox is under continuous development. We plan its development in three main directions.

- Implementation of new fuzzy model identification methods and techniques.
- Extending the existent implementations in order to support all kinds of polygonal membership functions.
- Implementation of new performance index types.

IV. CONCLUSIONS

Automatic fuzzy model identification from sample data is one of the key issues in practical application of fuzzy systems. After reviewing the methods ACP, RBE-DSS and RBE-SI this paper presented the RuleMaker toolbox a software tool that implements these methods. All the methods produce a fuzzy system with low complexity applying usually a sparse rule base. Therefore the created fuzzy systems should use a fuzzy rule interpolation based inference technique. The software seamlessly interoperates with the FRI Matlab toolbox. Thus the produced fuzzy systems are generated for one of the FRI inference techniques available in the FRI toolbox. The software is available under GNU GPL. One of the future researches will be focused on extending the approach to automatic fuzzy model identification methods for nonlinear plants control [31].

REFERENCES

[1] Baranyi, P., Kóczy, L. T. and Gedeon, T. D.: A Generalized Concept for Fuzzy Rule Interpolation, in IEEE Transaction on Fuzzy Systems, Vol. 12, No. 6, 2004. pp 820-837.

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

[3] Botzheim, J., Cabrita, C., Kóczy, L. T. and Ruano, A. E.: Fuzzy rule extraction by bacterial memetic algorithms, in Proceedings of

the 11th World Congress of International Fuzzy Systems Association, IFSA 2005, Beijing, China, July 2005, pp. 1563-1568.

[4] 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.

[5] Bouchon-Meunier, B., Marsala, C.; Rifqi, M.: Interpolative reasoning based on graduality. In Proc. FUZZ-IEEE'2000, 2000, pp. 483-487.

[6] Chiu, S. L.: Fuzzy Model Identification Based on Cluster Estimation, Journal of Intelligent and Fuzzy Systems, Vol. 2, 1994, pp. 267-278.

[7] Chong, A.: Constructing Sparse and Hierarchical Fuzzy Rulebases, PhD Thesis, Murdoch University, Perth, W.A., 2004.

[8] Chong, A., Gedeon, T. D. and Kóczy, L. T.: A Hybrid Approach for Solving the Cluster Validity Problem, in Proceedings of the 14th International Conference on Digital Signal Processing, Santorini, Greece, 2002, pp. 1207-1210.

[9] Chong, A., Gedeon, T. D. and Kóczy, L. T.: Projection Based Method for Sparse Fuzzy System Generation, in Proceedings of the 2nd WSEAS International Conference on Scientific Computation and Soft Computing, Crete, Greece, 2002, pp. 321-325.

[10] http://fri.gamf.hu A website dedicated to fuzzy rule interpolation and sparse fuzzy model identification issues.

[11] 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 Systems Symposium, 1989, pp. 247-250.

[12] GNU –General Public Licence: http://www.gnu.org/copyleft/gpl.html

[13] Johanyák, Z. C.: Fuzzy rule interpolation methods and automatic system generation based on sample data (in hungarian), PhD thesis, University of Miskolc, 2008.

[14] Johanyák, Zs. Cs. and Kovács, Sz.: Fuzzy Rule Interpolation by the Least Squares Method, in Proceedings of the 7th International Symposium of Hungarian Researchers on Computational Intelligence (HUCI 2006), Budapest, Hungary, 2006, pp. 495-506.

[15] Johanyák, Zs. Cs. and Kovács, Sz.: Sparse Fuzzy System Generation by Rule Base Extension, in Proceedings of the 11th IEEE International Conference of Intelligent Engineering Systems (IEEE INES 2007), Budapest, Hungary, 2007, pp. 99-104.

[16] Johanyák, Zs. Cs. and Kovács, Sz.: The effect of different fuzzy partition parameterization strategies in gradient descent parameter identification, in Proceedings of the 4th International Symposium on Applied Computational Intelligence and Informatics (IEEE SACI 2007), Timisoara, Romania, 2007, pp. 141-146.

[17] Johanyák, Zs. Cs., Tikk, D., Kovács, Sz. and Wong, K. W: Fuzzy rule interpolation Matlab toolbox – FRI toolbox, in Proceedings of the IEEE World Congress on Computational Intelligence (WCCI'06), 15th Int. Conf. on Fuzzy Systems (FUZZ-IEEE'06), 2006, Vancouver, BC, Canada, pp. 1427-1433.

[18] Klawonn, F. and Kruse, R.: Constructing a fuzzy controller from data, Fuzzy Sets and Systems, Vol. 85, Issue 2, 1997, pp. 177-193.

[19] L. T. Kóczy, J. Botzheim, and T. D. Gedeon. Fuzzy models and interpolation. In M. Nikravesh, J. Kacprzyk, and L. A. Zadeh, editors, Forging New Frontiers: Fuzzy Pioneers I & II, volume 217 of Studies in Fuzziness and Soft Computing. Springer, 2007.

[20] Kóczy, L. T. and Hirota, K.: Approximate reasoning by linear rule interpolation and general approximation, in International Journal of Approximative Reasoning, Vol. 9, 1993, pp. 197–225.

[21] Kóczy, L. T. and Hirota, K.: Size reduction by interpolation in fuzzy rule bases, in IEEE Transactions on System, Man and Cybernetics, Vol. 27, 1997, pp. 14-25.

[22] Kovács, Sz.: Extending the Fuzzy Rule Interpolation "FIVE" by Fuzzy Observation, Theory and Applications, Springer Berlin Heidelberg, 2006, pp. 485-497.

[23] Kosko, B. and Dickerson, J. A.: Function approximation with additive fuzzy systems, in Theoretical Aspects of Fuzzy Control, John Wiley & Sons, 1995.

[24] Kosko, B.: Optimal fuzzy rules cover extrema, International Journal of Intelligent Systems, Vol. 10, 1995, pp. 249 - 255.

[25] Sugeno, M. and Yasukawa, T.: A fuzzy-logic-based approach to qualitative modeling, in IEEE Transactions on Fuzzy Systems, Vol. 1, 1993, pp. 7-31.

[26] Tikk, D., Gedeon, T. D. Kóczy, L. T. and Bíró, G.: Implementation details of problems in Sugeno and Yasukawa's qualitative modeling, Research Working Paper RWP-IT-02-2001, School of Information Technology, Murdoch University, Perth, W.A., 2001.

[27] Weina Wang, Yunjie Zhang: On fuzzy cluster validity indices. Fuzzy Sets and Systems 158(19): 2095-2117 (2007)

[28] Wong, K. W., Gedeon, T. D. and Tikk, D.: An improved multidimensional α-cut based fuzzy interpolation technique, In Proc. Int. Conf. Artificial Intelligence in Science and Technology (AISAT'2000), Hobart, Australia, 2000, pp. 29–32.

[29] Wong, K. W., Kóczy, L. T., Gedeon, T. D., Chong, A. and Tikk, D. :Improvement of the Cluster Searching Algorithm in Sugeno and Yasukawa's Qualitative Modeling Approach, in Lecture Notes in Computer Science, Vol. 2206, 2001, pp. 536–549.

[30] Precup, R.-E. and Preitl, S.: PI-Fuzzy controllers for integral plants to ensure robust stability, Information Sciences, vol. 177, no. 20, Oct. 2007., , pp. 4410-4429.

[31] Precup, R.-E., Preitl, S., Tar, J. K., Tomescu, M. L., Takács, M., Korondi, P. and Baranyi, P.: Fuzzy control system performance enhancement by iterative learning control. IEEE Transactions on Industrial Electronics, Vol. 55, Nr. 9, 2008, pp. 3461-3475.