Johanyák, Zs. Cs., Kovács, Sz.:The effect of different fuzzy partition parameterization strategies in gradient descent parameter identification, 4th International Symposium on Applied Computational Intelligence and Informatics (SACI 2007), May 17-18, 2007 Timisoara, Romania, pp. 141-146.

The effect of different fuzzy partition parameterization strategies in gradient descent parameter identification

Zsolt Csaba Johanyák[1], Szilveszter Kovács[2]
Institute of Information Technologies
Kecskemét College, GAMF Faculty
Kecskemét, Hungary
[1]johanyak.csaba@gamf.kefo.hu, [2]kovacs.szilveszter@gamf.kefo.hu

***Abstract*—One of the most critical steps during the development of a fuzzy system is the identification of the fuzzy rule base and the fuzzy partitions, the so-called "tuning". This paper intends to present a comparative study of three different fuzzy partition parameter identification methods with respect to the effect of different fuzzy partition parameterization strategies.**

Keywords-fuzzy system tuning; rule base optimization; sparse rule base; fuzzy rule interpolation

I. INTRODUCTION

The automatic generation of the fuzzy rule base and the corresponding fuzzy partitions from a given set of input-output data is a very important part of the nowadays fuzzy systems research areas. There are numerous existing solutions for fuzzy rule base identification e.g. [7],[9], and [1].

In spite of the significant research work done in this field hardly can be found a comparative study of the different partition parameterization strategies. The main goal of this paper is to set up three standard fuzzy partition parameterization methods, and compare their efficiency by applying a gradient descent parameter identification method.

The rest of this paper is organized as follows. Section II. introduces three parameterization approaches and defines the constraints related to them. Section III. presents the performance index used for system evaluations and the tuning algorithm. The results of the experiments are analyzed in section III.

II. PARAMETERS AND CONSTRAINT DEFINITION

A. *The classical way (Method 1)*

The first approach we tried is identical with the parameterization used in [6] and [7]. Considering convex and normal trapezoidal shaped fuzzy sets the abscissas of the four break-points are selected as parameters.

The vertices are numbered in clockwise direction starting with left endpoint of the lower base. In course of the modification of the parameters the following constraints have to be applied.

- Starting from the second vertex the value of each parameter has to be greater or equal to its predecessor. It can be expressed by the inequality

$$p_k \geq p_{k-1}, 1 < k \leq 4, \tag{1}$$

where p_k is the current parameter, its subscript k indicates its position in the sequence of vertices.

- In case of the first three vertices the value of each parameter has to be smaller or equal to its successor. It can be expressed by the inequality

$$p_k \leq p_{k+1}, 1 \leq k < 4. \tag{2}$$

- The reference point, in this case the midpoint of the core has to be inside the range of the current linguistic variable. It can be expressed by the inequality

$$R_{\min} \leq \frac{p_2 + p_3}{2} \leq R_{\max}, \tag{3}$$

where $R_{\min}, R_{\max}$ are the lower respective upper endpoints of the range of the actual input/output dimension.

The last constraint is applicable only in case of inference methods that allow a fuzzy set situated at the margin of a range to lap over the touched boundary. The inference methods based on the concept of Linguistic Term Shifting (LTS) [4] belong to this category. Both of the techniques LESFRI [3] and FRIPOC [2], which were used in course of the preparation of this paper, apply LTS.

B. *Working with relative distances (Method 2)*

Another way of handling the vertices of a trapezoid results from transforming the original abscissa values into relative ones. In this case the first parameter serves for the identification of the position of the fuzzy set and it is equal to the horizontal co-ordinate of the left endpoint of the lower base of the trapezoid (see fig. 1). The position of the remaining three vertices is described by a positive or zero value indicating their

Research partly supported by the Kecskemét College, GAMF Faculty grant no: 1N076/2006.

distance from the predecessor break-point. Considering the same numbering convention as used in the previous section results the formula

$$r_k = \begin{cases} p_k, & k = 1 \\ p_k - p_{k-1}, & 1 < k \le 4, \end{cases} \tag{4}$$

where r_k denotes the k^{th} relative parameter. Hereby the constraints applicable during the modification of the parameters can be formulated simpler than in the case of the traditional parameterization way.

- The parameters except the first one have to be positive or zero.

$$r_k \ge 0, \quad 1 < k \le 4 \tag{5}$$

- The midpoint of the core has to be inside the range of the current linguistic variable. It can be expressed by the inequality system

$$R_{\min} \le r_1 + r_2 + \frac{r_3}{2} \le R_{\max}, \tag{6}$$

Similar to the case of the approach presented in section *II.A* the second constraint has to be fulfilled only when one uses a special group of fuzzy reasoning techniques.

C. Conserving the Ruspini character of the partition (Method 3)

A significant advantage of the automatically generated and tuned fuzzy systems against trained neural networks is the interpretability of the generated rules. This feature can be best exploited when the system is built up from Ruspini partitions, i.e. in each point the sum of membership values is equal to 1.

Here only the endpoints of the cores are used as parameters, the abscissas of the other two vertices are identical with the horizontal co-ordinates of the respective core-endpoints of the neighboring linguistic terms. In order to facilitate the application of LTS based fuzzy reasoning methods the first vertex of the leftmost set and the last (4^{th}) vertex of the rightmost linguistic term is determined by mirroring the other flank around a vertical axis crossing the midpoint of the core

$$x_{i,1} = \begin{cases} x_{i,2} - \left(x_{i,4} - x_{i,3}\right), & i = 1 \\ x_{i-1,3}, & 1 < i \le n, \end{cases} \tag{7}$$

$$x_{i,4} = \begin{cases} x_{i,2} - \left(x_{i,4} - x_{i,3}\right), & i = 1 \\ x_{i-1,3}, & 1 < i \le n, \end{cases} \tag{8}$$

where $x_{i,j}$ is the abscissa of the j^{th} break-point of the i^{th} fuzzy set.

Also in this case one has to define some constraints in order to ensure the validity of the sets and the desired character of the partition. They are the followings.

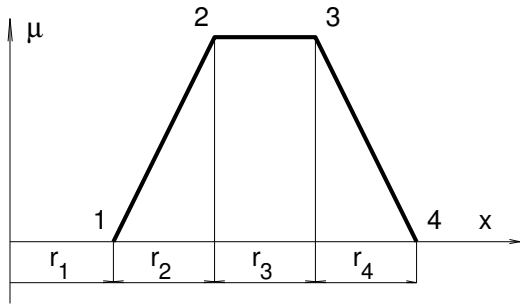

Figure 1. Relative distances as parameters

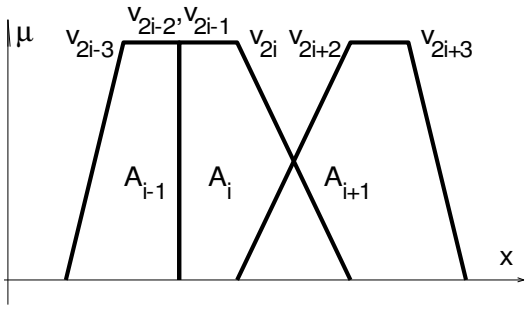

Figure 2. Two touching linguistic terms (A_{i-1} and A_i)

- The abscissa of the third vertex should have a greater or equal value to the abscissa of the second vertex. Supposing a vector representation of the parameters of all fuzzy sets in a partition the current condition can be described by

$$v_{2i-1} \le v_{2i}, \quad 1 \le i \le n \tag{9}$$

where n is the number of fuzzy sets in the actual partition, v is the vector containing the abscissas of the sets, and i is the ordinal number of the current linguistic term, for which the condition is formulated.

- The core of a set cannot even partly overlap the core of another linguistic term. However, two neighboring sets can touch each other (see fig. 2). In case of the i^{th} set this condition can be expressed by the following two inequalities

$$v_{2i-1} \ge v_{2i-2}, \tag{10}$$

$$v_{2i} \le v_{2i+1}. \tag{11}$$

Since the used parameter identification method modifies the parameters one at a time the above defined two constraints can be merged in one inequality system

$$v_{k-1} \le v_k \le v_{k+1}, \quad 1 < k < 2n \tag{12}$$

where k is the ordinal number of the currently adjusted parameter.

- The left endpoint of the first (leftmost) set cannot be part of the open interval of the range of the linguistic variable

$$v_1 \le R_{\min} . \tag{13}$$

- The right endpoint of the last (rightmost) set cannot be part of the open interval of the range of the linguistic variable

$$v_{2n} \ge R_{\max} . \tag{14}$$

- The midpoint of the core has to be inside the range of the current linguistic variable. It can be expressed by the inequality system

$$R_{\min} \le \frac{v_{2i-1} + v_{2i}}{2} \le R_{\max} . \tag{15}$$

Identical to the other two approaches condition (15) is applicable in case of parameter identification for LTS based inference methods.

III. Parameter Identification

A. Performance Index

In course of the parameter identification process after each parameter adjustment the resulting parameter set is evaluated by calculating the system output for a collection of predefined input data, for which the expected output values are known. In order to compare the results obtained different parameter sets a performance index is calculated after each system evaluation.

Both of the methods presented in [7] and [6] apply the mean square error (16) as performance index of a given parameter value set.

$$PI = \frac{\sum_{j=1}^{M} \left(y_j - \hat{y}_j\right)^2}{M}, \tag{16}$$

where M is the number of training data points, y_j is the output of the j^{th} data point and $\hat{y}_j$ is the output calculated by the system.

Contrary to their approach our algorithm uses the root mean square [8] (quadratic mean) of the error as performance index for the evaluation of the fuzzy system. We chose it owing to its better comprehensibility and comparability to the range of the output linguistic variable. Its value is calculated by

$$RMSE = \sqrt{\frac{\sum_{j=1}^{M} \left(y_j - \hat{y}_j\right)^2}{M}}, \tag{17}$$

The relative value of *RMSE* to the range (*RMSEP*) expressed in percentage is also monitored during the calculations

$$RMSEP = \frac{RMSE}{DR} \cdot 100, \tag{18}$$

where DR is the range of the output dimension. The application of *RMSEP* as performance index makes possible the parameter identification by our algorithm even in case of Multiple Input Multiple Output (MIMO) systems. In that case the resulting performance index is calculated using the formula

$$RRMSEP = \sqrt{\sum_{l=1}^{n_{out}} RMSEP_l^2}, \tag{19}$$

where and n_{out} is the number of the output dimensions.

B. Tuning Algorithm

The parameter identification method used in course of the calculations is a heuristic algorithm a variant of the gradient descent method like the technique presented in [7]. Contrary to the method suggested in [7] our algorithm modifies the parameters of the output linguistic terms as well. The parameters of the final system considered as optimal (corresponding to a local or global minimum of RMSE) are iteratively approximated by the algorithm introduced in fig. 3.

In course of iteration each parameter is modified one by one in both of the possible upper and lower (increasing and decreasing) directions. After determining a new value for the current parameter the system is evaluated calculating the actual value of the performance index. If this is better than the previous minimum the new parameter value is stored.

The amount of modification of the parameters is dependent on the range of the current input/output dimension, i.e. the step is calculated by multiplying the range by a coefficient. Reference [6] suggests a constant value 0.05 (5%) for this task. The authors of [7] propose a formula that starts the coefficient with a value of 0.0417 (4.17%) and increases it when the reduction of the performance index is smaller than 10% in course of an iteration stage.

We propose another adaptive approach, which determines the actual value of the coefficient depending on the change of RMSE during an iteration stage, the history of previous iteration stages, and a prescribed minimum value for the coefficient (C_{min}). For the calculation of C_{min} we determine first the range of each linguistic variable

$$DR_j = R_{j\max} - R_{j\min}, \quad 1 \le j \le n_{in} + n_{out}, \tag{20}$$

where n_{in} is the number of the input dimensions, $R_{j\min}$ and $R_{j\max}$ denote the lower respective upper endpoints of the current (j^{th}) dimension. The threshold for the coefficient is

$$C_{\min} = \frac{10^{-dn}}{\min\limits_{j \in [1, n_{in}+n_{out}]} \left(DR_j\right)}, \tag{21}$$

where *dn* is the maximal number of decimals used by the description of the parameters.

The iteration starts with a prescribed value of the coefficient (default: 0.2). If the improvement of the system slows down or even stops, i.e. the value of RMSE does not reduce more than a prescribed threshold (default: 0.001) during one iteration, the coefficient is divided by two unless its vale is already equal to C_{min}. In that case we generate a random value between 0 and 0.5 for the coefficient. It gives a chance to get out from the local minimum. We also limit the number of randomly generated coefficients (default: 2). Reaching this limit the iteration stops.

On the other hand the coefficient is increased multiplying it by two when the improvement of the system speeds up, i.e. the value of RMSE increases more than a specified threshold (default: 10) during one iteration. The algorithm also stops when the prescribed number of iteration is done. The whole algorithm is presented on fig. 3.

```
Evaluate the raw system with the training data
Store RMSE as RMSE minimum
Do iteration until the maximal number of iterations is reached
    For each input and for the output dimension
        Calculate modification step
        For each parameter being adjusted
            For both modification directions
                Calculate the modified value of the parameter
                Apply constraints
                Evaluate the system with the training data
                If RMSE actual < RMSE minimum
                    Store adjusted parameter value
                    Store RMSE actual as RMSE minimum
                EndIf
            EndFor
        EndFor
    EndFor
    If RMSE did not change in course of one iteration
        If Coefficient is equal to the minimum allowed value
            If the number of randomly generated coefficients
                is equal to the maximal allowed value
                    Stop iteration
            Else
                Generate a random Coefficient
            EndIf
        Else
            Halve Coefficient
        EndIf
    EndIf
    If RMSE decreased more than a threshold
        Double Coefficient
    EndIf
End
```

Figure 3. Tuning algorithm

IV. System Tuning Results

A. Introduction

In course of the examination of the parameterization methods we tuned two fuzzy systems, a Single Input Single Output (SISO) and a Multiple Input Multiple Output (MISO) one. The data used were generated using the function

$$y = 2 \cdot x + 1, \quad x \in [0, 10] \tag{22}$$

in the case of the SISO (first) system, and the function

$$y = \left(1 + x_1^{-2} + x_2^{1.5}\right)^2 \quad x \in [0.8, 1.8] \tag{23}$$

in the case of the MISO (second) system. The raw systems were generated by a similar technique to the method introduced in [1], which is also based on the clustering of the output and input spaces.

The first system contains 9 input and 3 output linguistic terms and 9 rules. The raw system ensures a full coverage of the input universe. Its dense character did not change during the training process. 101 data points were used for the generation and tuning of the system.

The second system contains 8 respective 9 linguistic terms in its two input dimensions, and its output partition is built up from 4 fuzzy sets. In this case the rule base is sparse, and it contains 18 rules. 196 data points were used for the generation and tuning of the system.

The systems were trained to the fuzzy rule interpolation based inference techniques FRIPOC [2] and LESFRI [3] separately. In course of the tuning process both of the RMSE (17) and RMSEP (18) were monitored. The time need for one iteration is different in case of the first two and the third parameterization method because as long as by parameterization *II.A* (Method 1) and *II.B* (Method 2) four parameters have to be adjusted for each fuzzy set, in case of parameterization *II.C* (Method 3) the number of adjustable parameters is only two. In order to ensure the comparability of the methods the horizontal axis indicates the number of system evaluations (SE) on figures showing the variation of the performance in course of the tuning process (see e.g. fig. 5). During a system evaluation one calculates the output for all values belonging to the training data set, and calculates both of the performance indexes RMSE and RMSEP.

B. Tuning the First System

In case of the first system (see fig. 5-6) clearly the *II.A* (Method 1) parameterization way ensured the best results by both inference methods.

In case of the first two parameterization approaches the better results are coupled with the jumbling of the linguistic terms on the antecedent side. Fig. 4 shows a graphic representation of the rule base after tuning with parameterization *II.A* for reasoning technique FRIPOC. The axes *x* and *y* correspond to the antecedent respective consequent universes, and the vertical axis represents the membership values. It is clearly visible that the consequent

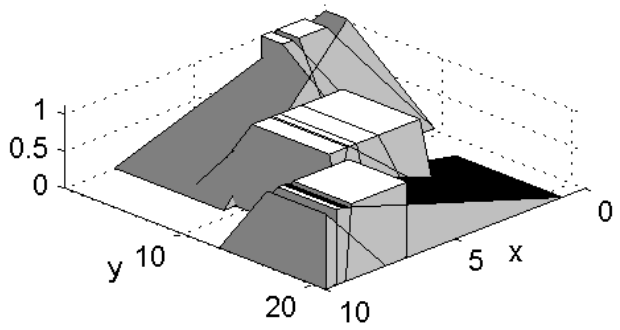

Figure 4. Rule base obtained after tuning the first system for the reasoning technique FRIPOC applying the *II.A* parameterization method

partition conserved its easy interpretability whilst the antecedent side contains several overlapping linguistic terms.

In contrast with this result in case of applying parameterization *II.C* (see fig. 7) both the consequent and the antecedent partitions preserve their interpretability thanks to the forced Ruspini type partitioning. However, the later approach leads to a weaker approximation capability of the system.

Using LESFRI as reasoning technique, in some stages of the tuning process parameterization *II.B* (Method 2) ensured better *RMSE(P)* values but at the end of the tuning process Method 1 was the winner. In some parts of fig. 6 the curves intertwined. Therefore we used different line width values in order to ensure their distinguishability.

C. Tuning the Second System

In case of the second (MISO) fuzzy system (see fig. 8-9) parameterization approach *II.B* led to the best results by both of

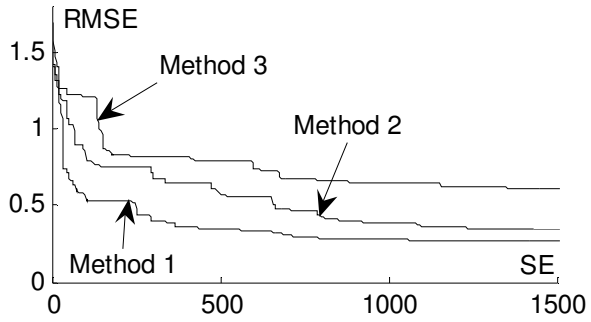

Figure 5. RMSE in course of tuning the first system for FRIPOC

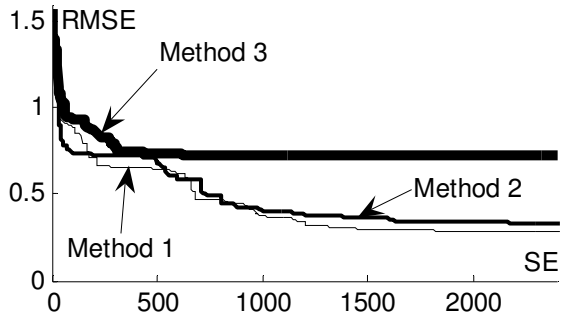

Figure 6. RMSE in course of tuning the first system for LESFRI

Figure 7. Rule base obtained after tuning the first system for the reasoning technique FRIPOC applying the *II.C* parameterization method

the inference techniques FRIPOC and LESFRI. However, the difference between the results gained with *II.A* and *II.B* was very small in case of LESFRI.

Similar to the previous experiments if one takes a view of figure 10., which shows the input and output partitions of the best performing system tuned for FRIPOC with *II.B*, it can recognize clearly the same phenomena of jumbling linguistic terms on the antecedent side as a result of the application of one of the first two parameterization ways.

Likewise it appeared in the case of the SISO system, the price of conserving the Ruspini character of the partition in all dimensions, which ensures the extraction of easily understandable rules, was paid by the performance of the system. Fig. 9 shows that the best RMSE value obtained in case of *II.C* is approximately twice as much than in the case of the parameterization method 1 and 2.

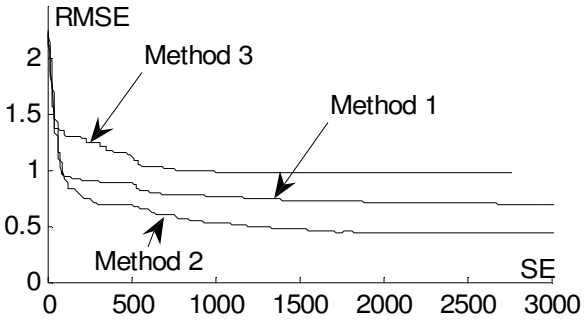

Figure 8. RMSE in course of tuning the second system for FRIPOC

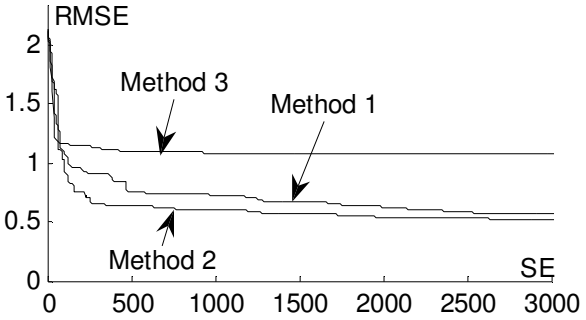

Figure 9. RMSE in course of tuning the second system for LESFRI

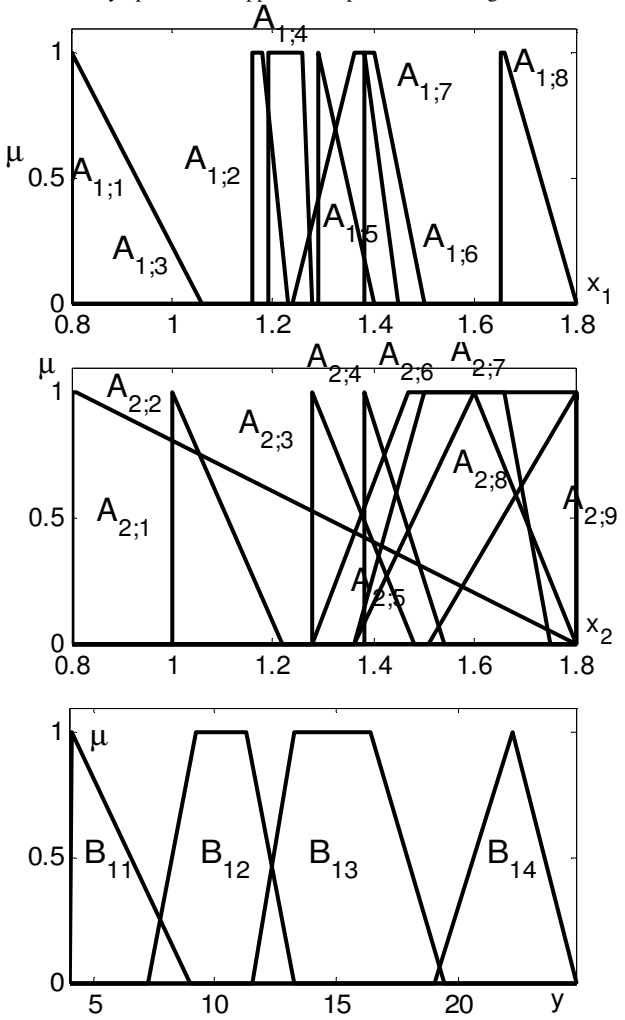

Figure 10. Input and output partitions of the best performing MISO system after tuning for FRIPOC with parameterization *II.B*

V. CONCLUSIONS

Parameter identification is a key step in fuzzy system development. This paper presented a comparative study of three parameterization approaches in order to find out the effect of parameterization method selection on the attainable system performance and on the decrease speed of the root mean square error used as performance index.

Two system types and two inference techniques were tried in course of the experimentation. In case of the SISO system the first parameterization method ensured the best result. In case of the MISO system the best performance index was encountered by the second parameterization approach. In both cases the system based on reasoning method FRIPOC [2] had the better tuning capability.

Although the third parameterization way has the great advantage of ensuring the extraction of always well interpretable rules it has been shown that this approach leads to a lower system performance by the applied parameter identification algorithm (see fig. 3). The enhancement of the algorithm in this direction is subject for further research work.

The implementation in Matlab of the presented tuning methods can be downloaded from [10]. This website is dedicated to a fuzzy rule interpolation Matlab toolbox development project (introduced in [5]) aiming the implementation of various FRI techniques.

REFERENCES

[1] A. Chong, "Constructing Sparse and Hierarchical Fuzzy Rulebases," PhD Thesis, Murdoch University, 2004.

[2] Z. C. Johanyák and S. Kovács, "Fuzzy Rule Interpolation Based on Polar Cuts," Computational Intelligence, Theory and Applications, Springer Berlin Heidelberg, 2006, pp. 499-511.

[3] Z. C. Johanyák and S. Kovács, "Fuzzy Rule Interpolation by the Least Squares Method," 7th International Symposium of Hungarian Researchers on Computational Intelligence (HUCI 2006), November 24-25, 2006 Budapest, pp. 495-506.

[4] Z. C. Johanyák and S. Kovács, "Fuzzy set approximation using polar co-ordinates and linguistic term shifting," SAMI 2006, 4th Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence, Herl'any, Slovakia, January 20-21 2006, pp. 219-227.

.[5] Z. C. Johanyák, D. Tikk, S. Kovács, K. W. Wong "Fuzzy Rule Interpolation Matlab Toolbox – FRI Toolbox," Proc. of the IEEE World Congress on Computational Intelligence (WCCI'06), 15th Int. Conf. on Fuzzy Systems (FUZZ-IEEE'06), July 16--21, 2006, Vancouver, BC, Canada, pages 1427-1433, Omnipress.

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

[7] D. Tikk, T. D. Gedeon, L. T. Kóczy and G. Bíró, "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.

[8] E. W. Weisstein, "Root-Mean-Square," From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/Root-Mean-Square.html

[9] 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 Approach," Lecture Notes in Computer Science, vol. 2206, pp. 536–549, 2001.

[10] http://fri.gamf.hu