A New Algorithm for the Design of Mamdani-type Fuzzy Controllers

Jean J. Saade
ECE Department, FEA
American University of Beirut, Lebanon
e-mail:jsaade@aub.edu.lb

Summary

This paper presents a new data-driven algorithm for the design of Mamdani-type fuzzy controllers. It is based on a new and parametrized defuzzification strategy. The algorithm is tested for data-fitting, function shape representation, noise insensitivity and generalization capability.

Keywords: Fuzzy controllers; learning; design algorithm; defuzzification.

1. Introduction

Most of the recent research in the design of fuzzy controllers has been centered on automatic techniques using expert's input-output data. The majority of these techniques rely on the use of Takagi-Sugeno type controllers and fuzzy-neural-network [3-5,11], fuzzy clustering and fuzzy partition approaches [1,2,6,7]. These controllers, however, are not fully-linguistic and the use of neural-network and other learning algorithms to tune initial controllers could lead to final systems that are difficult to interpret linguistically [4,6,7].

In a previous study, the author of this research paper developed a new and parametrized defuzzification strategy [8]. This strategy was shown in [9] to possess important properties. Of particular importance is the containment of a free parameter, which can be used to modify the crisp defuzzified values to fit the problem of concern. In this study, the above-noted defuzzification method is used to introduce a new data-driven algorithm for the design of Mamdani-type or fully-linguistic fuzzy controllers.

2. Defuzzification and design algorithm

Consider an N-rule, two-input, one-output fuzzy inference system. Let the j-th rule, $1 \le j \le N$, be:

If $x1$ *is* Aj *and* $x2$ *is* Bj, *then* z *is* Cj .

$x1$, $x2$ and z are the input and output variables of the system. Also, Aj, Bj and Cj are fuzzy sets defined respectively over $x1$, $x2$ and z. The output fuzzy set, corresponding to some crisp input pair $(x11, x21)$, can be obtained using the CRI [12] as follows:

$$C_{01}(z) = \max_{1 \le j \le N} [Aj(x11) \wedge Bj(x21) \wedge Cj(z)]. \qquad (1)$$

The fuzzy OR, AND and implication (FI) are respectively represented by maximum, minimum and minimum.

Now, the new defuzzification method applies to the normalized version of $C_{01}(z)$, denoted $C_{01n}(z)$, and obtained by dividing the membership function of $C_{01}(z)$ by its highest membership grade, as follows:

$$F_\delta [C_{01n}(z)] = \int_0^1 [\delta\, c_1(\alpha) + (1-\delta)\, c_2(\alpha)]\, d\alpha\,. \qquad (2)$$

$[c_1(\alpha), c_2(\alpha)]$ is the α-level set of C_{01n} and δ is a parameter that takes values in [0,1]. Eq.(2) was established by the author in [8,10].

It is assumed that the system designer is able to specify the input and output variables of the fuzzy controller and the ranges of these variables. Then overlapping membership functions are assigned to cover the entire ranges of the variables of concern. In terms of overlap, a related study [9] recommended that the input membership functions, assigned over a single input variable, be such that the membership grades of any crisp input value sum to 1. This was shown to help achieve smoothness and continuity of the controller input-output surface. Also, the use of the sum-product-product for OR-AND-FI was shown to serve the same objectives more than other combinations.

Once the input membership functions are assigned, then all combinations of input fuzzy sets are considered to form the antecedent parts of the rules. The initial rules consequents need to be equal to the left-most output fuzzy set. This set is required to be formed by a flat and a decreasing part or a decreasing part only. In the same manner, the right-most output fuzzy set is to be formed by a flat and an increasing part or an increasing part only.

Given the input-output data pairs in the form $(\underline{xi}, zid)$, with i=1, 2, 3,…,n and $\underline{xi} = (x1i, x2i, x3i, ..., xpi)$, where p is the number of input variables, the learning process starts with an initial fuzzy system as specified

above. The algorithm, whose flow-chart is shown in Fig. 1, computes the output fuzzy sets for all $\underline{xi}$ using Eq. (1) and then defuzzifies their normalized versions using Eq. (2) when δ =1. Here, all the defuzzified values will be equal to the smallest value of the output range. Hence, given that zid are all greater than or equal to this value (this should always be the case), then $F_1[C_{0in}(z)] \le zid$. For these defuzzified values, the error E is computed using some error function and compared with a desired error value E_d. If $E \le E_d$, then the learning stops. Otherwise, δ is decreased from 1 to 0 by passing through

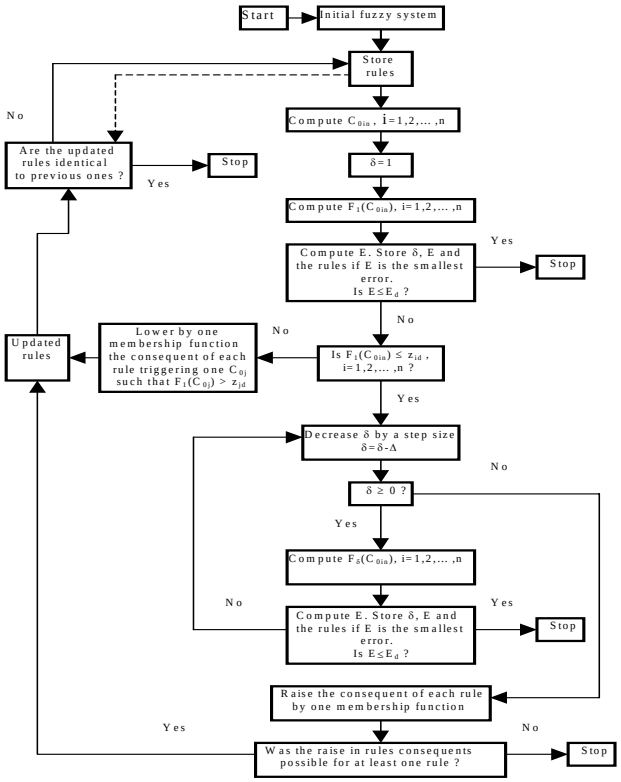

Figure 1. Flow-chart of the design algorithm.

discrete intermediate values. For each δ, the error is computed and compared with E_d. Note here that the decrease in δ results in an increase in the defuzzified values of the output fuzzy sets. These values could then be made closer to the desired outputs. Whether the change in δ has led to the satisfaction of the error goal, that is, $E \le E_d$ has been achieved for some $\delta \in [0,1]$, then the learning stops. Otherwise, the algorithm starts again from $\delta = 1$ but with a new set of rules.

This new set of rules is obtained by raising each rule consequent by one fuzzy set. This, however, might lead to a violation of the inequality $F_1[C_{0in}(z)] \le zid$. If so, it can be reestablished by lowering the consequents of the rules which trigger output fuzzy sets whose defuzzified values for $\delta = 1$ are greater than their desired counterparts. Once all defuzzified values become again smaller than or equal to the desired ones, then δ will be decreased from 1 to 0 and for each δ the error is computed and compared with E_d. This process is repeated until either the error requirement is satisfied or no more raise in the rules consequents is possible or when the raise and lowering of the rules consequents result in a system that has already been obtained. When the learning stops, the algorithm delivers the final fuzzy system with the least error value that can be obtained under the described procedure, the error and the final δ value.

3. Example

In this example, the following two-variable non-linear function is considered:

$$z = f(x1, x2) = (1 + x1^{-2} + x2^{-1.5})^2, 1 \le x1, x2 \le 5. \quad (3)$$

The input-output surface of this function is shown in Fig. 2. This function was first considered in [7] and then considered again in [2] and [6]. 50 input-output data points (listed in [7] and in Table 1) were considered in the noted references.

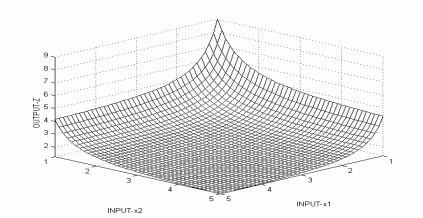

Figure 2. Input-output surface of the function given in Eq. (3).

In [7], a 6-rule fuzzy system was determined with 0.318 as a mean-square error (MSE) value. The error was then reduced to 0.01 using position gradient. In [2], which is also a study based on fuzzy clustering, the best obtained MSE was 0.231. 5 rules were considered. The use of fuzzy partition in [6] gave a 6-rule fuzzy system with 0.351 as a MSE value. This was then reduced to 0.005 by a fuzzy neural network.

Table 1: 50 training data points used in the Example given in Section 3.

#	x1	x2	z	#	x1	x2	z
1	1.4	1.8	3.7	26	2	2.06	2.52
2	4.28	4.96	1.31	27	2.71	4.13	1.58
3	1.18	4.29	3.35	28	1.78	1.11	4.71
4	1.96	1.9	2.7	29	3.61	2.27	1.87
5	1.85	1.43	3.52	30	2.24	3.74	1.79
6	3.66	1.6	2.46	31	1.81	3.18	2.2
7	3.64	2.14	1.95	32	4.85	4.66	1.3
8	4.51	1.52	2.51	33	3.41	3.88	1.48
9	3.77	1.45	2.7	34	1.38	2.55	3.14
10	4.84	4.32	1.33	35	2.46	2.12	2.22
11	1.05	2.55	4.63	36	2.66	4.42	1.56
12	4.51	1.37	2.8	37	4.44	4.71	1.32
13	1.84	4.43	1.97	38	3.11	1.06	4.08
14	1.67	2.81	2.47	39	4.47	3.66	1.42
15	2.03	1.88	2.66	40	1.35	1.76	3.91
16	3.62	1.95	2.08	41	1.24	1.41	5.05
17	1.67	2.23	2.75	42	2.81	1.35	1.97
18	3.38	3.7	1.51	43	1.92	4.25	1.92
19	2.83	1.77	2.4	44	4.61	2.68	1.63
20	1.48	4.44	2.44	45	3.04	4.97	1.44
21	3.37	2.13	1.99	46	4.82	3.8	1.39
22	2.84	1.24	3.42	47	2.58	1.97	2.29
23	1.19	1.53	4.99	48	4.14	4.76	1.33
24	4.1	1.71	2.27	49	4.35	3.9	1.4
25	1.65	1.38	3.94	50	2.22	1.36	3.39

As for the algorithm described in this study, the same 50 data points were considered. As in Figs. 3(a) and

3(b), 3 membership functions were considered over each of the input variables. Hence, 9 inference rules were adopted. The best error value of 0.216 was obtained with 4 output fuzzy sets (Fig. 3(c)). The final obtained fuzzy rules are as in Eq. (4) and the final δ value is 1.The input-output surface is shown in Fig. 4.

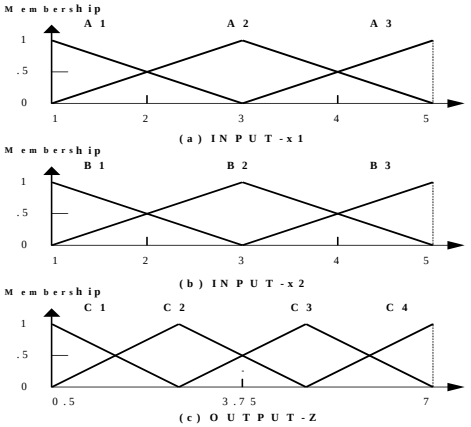

Figure 3. Membership functions assigned over the input and output variables of the function given in Eq. (3).

If x1 is A1 and x2 is B1, then z is C3
If x1 is A1 and x2 is B2, then z is C3
If x1 is A1 and x2 is B3, then z is C3
If x1 is A2 and x2 is B1, then z is C3
If x1 is A2 and x2 is B2, then z is C2
If x1 is A2 and x2 is B3, then z is C2
If x1 is A3 and x2 is B1, then z is C3
If x1 is A3 and x2 is B2, then z is C2
If x1 is A3 and x2 is B3, then z is C2 (4)

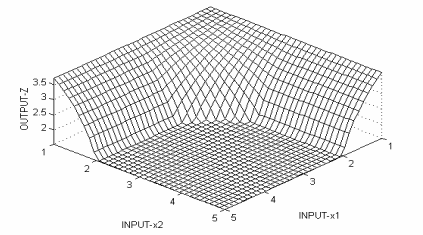

Figure 4. Input-output surface of the fuzzy system in Eq. (4)

Hence, the presented results show that the proposed algorithm can provide a better data-fitting than the clustering and fuzzy partition methods [2,6,7]. The data-fitting, however, is inferior to that of the combined fuzzy-partition-neural-network [6] and clustering-gradient-position [7]. Knowing that a smaller error value obtained in the approximation of data does not necessarily imply a better function shape representation, and since the surfaces of the obtained fuzzy systems in [2,6,7] were not given, we choose to compare the results of this example with those obtained by ANFIS [4]. It is available under a MATLAB tool-box. Also the presented approach will be tested for noise insensitivity and generalization (extrapolation). ANFIS is not structured to account for this type of generalization.

The use of the same 50 data pairs (Table 1) and 3 triangular membership functions over each of the 2 input variables in ANFIS, which is based on a combined gradient-descent least-squares, gave a nine-rule fuzzy system with 0.0303 as an MSE under 100 epochs and 0.00001 initial step-size. The input-output surface is shown in Fig. 5. These are the best obtained results. Since the error value is smaller than 0.216 obtained in our approach, then ANFIS provided a better data-fitting. The comparison of Figs. 4 and 5, however, reveals that the proposed approach has a better representation of the shape of the non-linear function.

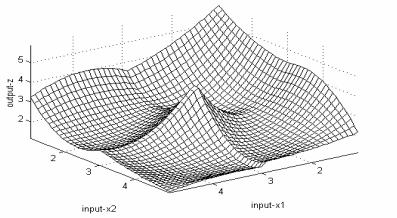

Figure 5. Input-output surface of the 9-rule fuzzy system obtained from ANFIS using the 50 data pairs in Table1.

Concerning the noise insensitivity issue, we considered 3 stages of modification of output values in the 50 points listed in Table 1. In each stage, 4 output values were modified to result in 4 input-output pairs, denoted as a set, not satisfying the function in Eq. (3) (see Table 2). First, set 1 was used in addition to the remaining 46 noise-free data pairs. In stage 2, sets 1 and 2 were used in addition to the remaining 42 noise-free data pairs. In stage 3, sets 1, 2 and 3 were used in addition to the remaining 38 noiseless data. The 9-rule fuzzy systems obtained by ANFIS had 0.0422, 0.0415 and 0.1037 MSE and surfaces shown in Figs. 6(a), 6(b) and 6(c) respectively.

Table 2: 50 noisy and noise-free data points.

#	x1	x2	z		#	x1	x2	z	
1	1.4	1.8	3.7		26	2	2.06	2.52	
2	4.28	4.96	1.31		27	2.71	4.13	1.58	
3	1.18	4.29	3.35		28	1.78	1.11	4.71	
4	1.96	1.9	2.7		29	3.61	2.27	1.87	
5	1.85	1.43	3.52		30	2.24	3.74	2.8	**
6	3.66	1.6	2.46		31	1.81	3.18	3	**
7	3.64	2.14	1.95		32	4.85	4.66	1.3	
8	4.51	1.52	4	*	33	3.41	3.88	1.48	
9	3.77	1.45	2.7		34	1.38	2.55	3.14	
10	4.84	4.32	1.33		35	2.46	2.12	2.22	
11	1.05	2.55	4.63		36	2.66	4.42	1.56	
12	4.51	1.37	4.2	*	37	4.44	4.71	1.32	
13	1.84	4.43	2.7	**	38	3.11	1.06	4.08	
14	1.67	2.81	2.47		39	4.47	3.66	1.42	
15	2.03	1.88	3.7	***	40	1.35	1.76	3.91	
16	3.62	1.95	2.08		41	1.24	1.41	5.05	
17	1.67	2.23	2.75		42	2.81	1.35	1.97	
18	3.38	3.7	1.51		43	1.92	4.25	3	**
19	2.83	1.77	3.7	***	44	4.61	2.68	3.6	*
20	1.48	4.44	2.44		45	3.04	4.97	1.44	
21	3.37	2.13	1.99		46	4.82	3.8	3.1	*
22	2.84	1.24	4.1	***	47	2.58	1.97	2.29	
23	1.19	1.53	4.99		48	4.14	4.76	1.33	
24	4.1	1.71	2.27		49	4.35	3.9	1.4	
25	1.65	1.38	3.94		50	2.22	1.35	4	***

* denotes set 1, ** denotes set 2 & *** denotes set 3.

All the above-noted 3 cases of noisy and noise-free data pairs were entered into the algorithm proposed in this study. The resulting 9-rule fuzzy system was always as in Eq. (4) and with δ=1. The resulting error values were respectively 0.3842, 0.4583 and 0.5056. The input-output surface is just the one shown in Fig. 4. The comparison of Fig. 4 with Figs. 6(a), (b) and (c) reveals that the presented approach has a better noise insensitivity than ANFIS. This result is also supported by the error values at

the noisy points. In terms of the error obtained by considering the original 50 noise-free data (Table 1), ANFIS gave 0.2292, 0.2925 and 0.3217 respectively. Comparison of these error values with 0.216 (obtained using the presented methodology), and also Fig. 4 with Figs. 6(a), (b) and (c), shows that ANFIS has in this example an inferior performance efficiency when the learning is done based on noisy data. This efficiency is measured by considering the noise-free data error and function shape representation.

In terms of testing the generalization capability of the presented approach, sets of data points from among those listed in Table 1 and located in specific regions were excluded in succession. The remaining data were entered into the presented algorithm. First, data pairs such that $1< x_1 <2.5$ and $3.5< x_2 <5$ were eliminated. This resulted in the exclusion of 5 data points. Second, data points such that $1< x_1 <3$ and $3< x_2 <5$ (8 points) were excluded. In both cases, the final fuzzy system turned out to be as in Eq. (4) and $\delta=1$. The error values were respectively 0.2355 and 0.2468. In both cases, the error value 0.216 still holds for the original 50 points in Table 1.

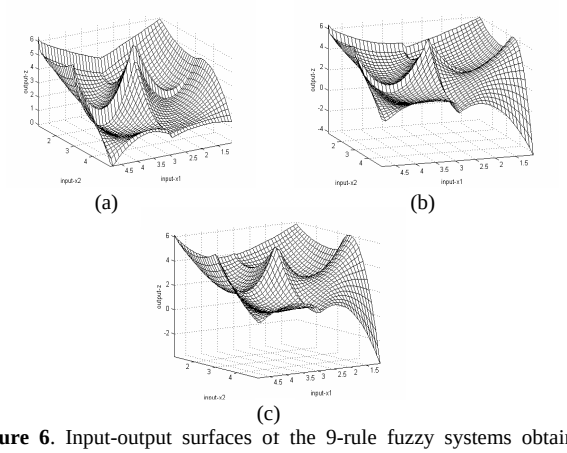

Figure 6. Input-output surfaces of the 9-rule fuzzy systems obtained from ANFIS.

The proposed algorithm was also tested for its ability to combat noise and generalize simultaneously. The data elimination process described in the preceding paragraph was again considered and noisy data from among the 12 data points indicated in Table 2 were introduced. The introduced noisy points were those which survived the elimination. Hence, 9 noisy points (set 1, 1 point from set 2 and set 3) were used among the 45 data pairs, which resulted from the first data exclusion process. Also, 8 noisy points (set 1 and set 3) were used among the 42 data pairs, which resulted from the second data elimination process. In both cases, the algorithm returned the final fuzzy system expressed in Eq. (4) with $\delta=1$. The input-output surface is therefore as shown in Fig. 4.

4. Conclusions

In this study, a new data-driven algorithm for the design of Mamdani-type and simple-to-read fuzzy controllers has been presented. It has been based on the use of a new and parametrized defuzzification strategy. The algorithm has been tested through the use of a typical non-linear function, which was considered in a number of published papers. In the case of noiseless data, the data-fitting of the proposed approach has been shown in this example better than the one pertaining to the fuzzy clustering and fuzzy partition. It has been worse, however, than the data-fitting capability of the combined fuzzy-partition neural-network, clustering gradient-position approaches and ANFIS. Yet, the function shape representation of the algorithm turned out to be better than ANFIS. When the data are noisy, a case that corresponds more to real situations, the presented approach has given a smaller noise-free data error, better noise insensitivity and function shape representation than ANFIS. In addition, the algorithm has been shown able to generalize and to combat noise and generalize simultaneously.

Acknowledgements
This research has been supported by the Research Board at the American University of Beirut.

References

[1] J.-Q. Chen, Y.-G. Xi and Z.-J.Zhang, A clustering algorithm for fuzzy model identification, Fuzzy Sets and Systems 98 (1998) 319-329.

[2] M. Delgado, A.F.G.–Skarmita and F.Martin, A fuzzy clustering-based rapid prototyping for fuzzy rule-based modeling, IEEE Trans. Fuzzy Systems, Vol. 5, No.2, (1997) 223-233.

[3] S.Horikawa, T.Furuhashi and Y.Uchikawa, On fuzzy modeling using fuzzy neural networks with the backpropagation algorithm, IEEE Trans. Neural Networks, Vol. 3, No. 5 (1992) 801-806.

[4] J.-S.R. Jang, ANFIS: Adaptive-network-based fuzzy inference system, IEEE Trans. Systems, Man and Cybernetics, Vol. 23, No. 3 (1993) 665-685.

[5] J.-S.R. Jang and C.-T. Sun, Neuro-fuzzy modeling and control, Proceedings of the IEEE, Vol. 83, No. 3 (1995) 378-406.

[6] Y.Lin, G.A. Cunningham III, and S.V. Coggeshall, Using fuzzy partitions to create fuzzy systems from input-output data and set the initial weights in a fuzzy neural network, IEEE Trans. Fuzzy Systems, Vol. 5, No. 4 (1997) 614-621.

[7] M. Sugeno and T. Yasukawa, A fuzzy-logic-based approach to qualitative modeling, IEEE Trans. on Fuzzy Systems, Vol. 1, No. 1, (1993) 7-31.

[8] J.J. Saade, A unifying approach to defuzzification and comparison of the outputs of fuzzy controllers, IEEE Trans. Fuzzy Systems, Vol. 4, No. 3 (1996) 227-237.

[9] J.J. Saade and H.B. Diab, Defuzzification methods and new techniques for fuzzy controllers, submitted to journal of Fuzzy Sets and Systems.

[10] J. J. Saade and H. Schwarzlander, Ordering fuzzy sets over the real line: An approach based on decision making under uncertainty, Fuzzy Sets and Systems 50 (1992) 237-246.

[11] H. Takagi and I. Hayashi, NN driven fuzzy reasoning, Int'l Journal of Approximate Reasoning, Vol. 5, No.3 (1991) 191-212.

[12] L. A. Zadeh, Outline of a new approach to the analysis of complex systems and decision processes, IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-3 (1973) 28-44.