Computer Science Journal of Moldova, vol.4, no.1(10), 1996

Optimization of fuzzy controllers by exhaustiv, trial & error and genetic methods. Medical applications

L.Boiculese H.Teodorescu G.Dimitriu

Abstract

The membership functions and rules optimization of a fuzzy controller are focused in this paper. Four optimization methods are addressed: (i) exhaustive method, (ii) trial and error, (iii) genetic algorithm, (iv) a combination of (ii) and (iii). The methods were tested in an application for anesthesia control, using a fuzzy controller with two inputs and one output, implemented on a PC. The results obtained by the classic method, by the trial-and-error (T.E.), by genetic algorithms (G.A.) and by a mixture of G.A. & T.E. are contrasted. The number and type of the membership functions (m.f.), are also taken in account. It is shown that a combination of G.A. & T.E. gives best results in rules deduction.
keywords: fuzzy control, exhaustive, trial and error, genetic algorithm, anesthesia control

1 Introduction

Fuzzy logic is often used in the field of bio-medical engineering (BME), see for instance [1,2,5,8,10] and in other fields [11], because of the advantages related to approximate reasoning and natural language implementation and processing [9]. Moreover, fuzzy systems have much in common with expert systems, because of the knowledge they can use and learn. This increases the potential applications in BME, where there is a large amount of knowledge which is heuristic.

To control a process means to read the information from it and to derive the new values of the control parameters. Using fuzzy control, this equals to fuzzify the inputs, to apply the rules and than to convert the fuzzy output control parameters, to crisp values.

The optimization of the fuzzy controller can be done by adjusting the fuzzification operator, by changing the rules table, and by choosing an appropriate defuzzification operation.

The simulation function for blood pressure control presented in [2,8] was used as a benchmark test. Of course, the presented system and algorithms can be applied in other applications too. Simulations regarding the control of other systems are also reported in this paper.

The aim of anesthesia controller is to keep under control the mean arterial blood pressure at a low value. This can be done by adjusting the anesthesia agent to keep the pressure constant, or with small variations from the desired set value. This drug has a vasodilator effect and so one can decrease the mean arterial blood pressure. For the fuzzy controller, the optimization is equivalent to find the optimum set of rules and membership functions to fulfill the above requirements. This control is also often used for postsurgical patients that have elevated blood pressure. The action of the drug is powerful and about 3-5 minutes after insertion. This implies the frequent pressure monitoring and vasodilator drug adjustment. Depending on the nurse experience, the adjustment is more or less accurate (with an acceptable error). Because of this, an automatic control loop is necessary.

The fuzzy control system is used in a feedback configuration, as pictured in fig.1.

The inputs are the error at the current moment and the change of error at the previous moment. The output represents the isoflurane concentration needed for patient inhale.

The discrete time function, that simulates the human body in this experiment, is the linear recursive function described below:

$$\begin{aligned} y(k) \;=\; & -a1 \cdot y(k-1) - a2 \cdot y(k-2) \\ & +b1 \cdot u(k-\tau 1) + b2 \cdot u(k-\tau 1 - 1) \\ & +b3 \cdot u(k-\tau 2) + b4 \cdot u(k-\tau 2 - 1) \end{aligned} \tag{1}$$

$a1 = -1.331,\ a2 = 0.335,\ b1 = 0.030,\ b2 = -0.048,$
$b3 = 0.017,\ b4 = -0.041,\ \tau 1 = 23[s],\ \tau 2 = 101[s]$

Obviously, the order of the controlled linear system alters the performance of the controller. Because of the biological diversity, the coefficients ai and bi depend on human sensitivity [5].

The membership functions number and type are also involved in the behavior of the system.

The system was tested for a big number of membership functions (from 2 to 10).

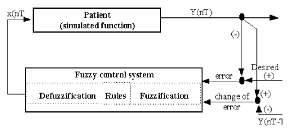

Figure 1. Block diagram of the control loop

2 Membership functions optimization

A part of the system knowledge is implemented in the input variables, using the fuzzification process. To best fit the knowledge in this operation the system was optimized using three types of membership functions.

The study was done for triangular, trapezoidal and Gaussian m.f.. Also the distance from the peaks of the m.f. the p-value, can be automatically (using trial and error) set up. The influence of p value upon m.f., can be view looking at figure 2.

For a complete definition of the Gaussian and trapezoidal m.f. the user has to input the τ — standard deviation, respective the ratio r between the length of the small base and large base.

An automatic method like **trial and error** is used to optimize these parameters: p, τ, r.

The method starts by increasing the value of the parameter, with a quantum depending on time. It goes with the increasing till the error is declining. If the error is growing, then the quantum is dividing by 2 and the procedure change the sense in optimizing the parameter, so the value is decreased with the new quantum. The procedure is stooped after 10 cycles.

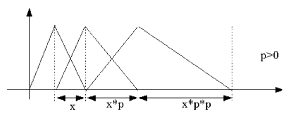

Figure 2. Membership function parameter p

3 Exhaustive method (rules optimization)

The system is tested for all possible set of rules combinations. A fuzzy system is characterized by a table of rules, which reflect the correspondence between the inputs and outputs. We use these notations:

- n_e error number of membership functions
- n_{ce} change of error number of membership functions
- n_o output fuzzy number of membership functions

The total number of possible sets of rules are:

$$N_t = n_o^{(n_{ce} \cdot n_e)} \tag{2}$$

Taking into account a number of $p = 2000$ steps of calculus, also $n_e = n_{ce} = n_o = 3$ number of m.f. and suppose that we have a powerful computer that needs only $t1 = 0.001\,s$ for one step of calculus, the total time needed for optimization is:

Ttotal=$3^9 * 2000 * 0.001\,(s) = 10.9$ hours

It is easy to see, that for a linear variation of membership functions number, the time of calculus is exponential increased, so that calculus time of years rank, are usually expected. This is the major drawback of this method, despite the benefit of global minimum error found.

To reduce the computation time, one can use a statistic method, such as Monte Carlo search of the optimum. However this method will not decrease enough the computation time. A batter choice could be the "trial – and – error" search method.

4 Trial and error method (rules optimization)

The system either starts with a set of rules proposed by the user, or it proposes the minimum output membership function (m.f.) number 1, for the conclusion of all the rules.

For each rule, the conclusion is increased (or decreased) one step, according to the "trial and error". It goes that way if the general error is diminished. The procedure can be easy understood having a look at Figure 3:

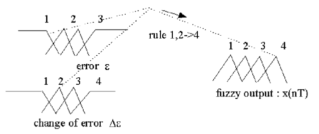

Figure 3. Example of a rule

For a new conclusion of the rule, the system calculates the current error. For this example, the test conclusion can be either: $4+1=5$, implying the chose of 1, or $4-1=3$ implying the chose of 3. The algorithm is (as in [4]):

0 initialize the set of rules

1 chose one rule; update the rules base

2 shift to next, or to the previous membership function (increase or decrease the number of the m.f. by 1) in the conclusion of that rule (trial and error)

3 if the error is diminishing, then continue with step 2

4 if there are rules not processed then go to 1 else STOP.

To avoid the locking of the algorithm in a given unacceptable state of the fuzzy system, a good starting set of rules has to be chosen. It is better for the behavior of the system if the user is a specialist in that field, a medical operator for example. The system can become trapped in a local minimum in the space of error. In the next tables such a case is presented:

Start set point
cycle state

	change of error			
error	1	2	3	4
1	4	4	4	4
2	4	4	4	4
3	4	4	4	4
4	4	4	4	4

the cycle state is achieved after two steps

minimum error

	change of error			
error	1	2	3	4
1	4	4	4	4
2	4	4	4	4
3	4	4	4	4
4	4	4	1	2

That's why the choice of the starting set of rules is important. If there are no local minima between the start set point and the global optimum, then the general minimum error is reached [2]. Because of this feature, we call this procedure the "smooth", or "soft" one.

5 Genetic algorithm method (rules optimization)

In the last decade genetic algorithms (GA) were successfully used in control systems optimization [1,3,6,7,12].

GA are search methods, that use the statistical technique. GA are inspired from the Darwinian principle of "survival of the fittest". They are well suited to cope with the traps of the local minima. GA are found to be very efficient in irregular and complex spaces in getting the global optimum of the problem.

Remember the elementary operations that GA uses. The *generation of the new population* is performed in order to ameliorate the value of a parameter. One searches in a collection of values, that which best matches the system function. The number of possible values defines the working population. For each element (individual) of the population, one can calculate a fitness function (that is the opposite of the error function at first glance). The newly population also has n individuals (chromosomes), but some of them are more than once generated till others disappear. Here, the chromosomes are the rules.

Below, the process of conversion of a set of rules in a chromosome is exemplified.

TABLE OF RULES				
		change of error		
	no.	1	2	3
	1	1	2	1
error	2	2	3	1
	3	3	3	2
	4	3	3	4

chromosome: 1 2 1 2 3 1 3 3 2 3 3 4

The fitness function is:

$$fitness = \frac{1}{\alpha \cdot \sum_j \left[y(j) - yd\right]^2} \tag{3}$$

j – represents the number of steps of calculus.
α – coefficient (0.3 for this experiment).
yd – desired value.

Generation of a new population using five m.f. for error; two m.f. for change of error; five m.f. for the vasodilator agent. P0 stands for the initial population.

Population PO					
no.	chromosome	fitness fi	$\frac{fi}{\sum_j fj}$	multiply by 4	integer
1	2223134333	5.611E-4	0.1664	0.665	1
2	5441324142	4.920E-4	0.1459	0.583	1
3	5413514135	2.39E-3	0.604	2.41	2
4	5133144212	2.786E-4	0.082	0.33	0
total:		3.71E-3	1≡100%	4	4

The new population P1, generated from P0, with the restriction of the fitness function is:

		chromosome	old position	
P1	1	2223134333	1	Some individuals
	2	5441324142	2	that are not
	3	5413514135	3	adapted will die.
	4	5413514135	3	(the element 4 from P0).

The *crossover* operation is used to produce off-springs, that inherit information computed and selected from the parents. By using these elements, new points in the space of works are reached. This process was realized with the probabilities in the interval [60%, 70%] for simulations, but there were no significant differences when choosing the probability in this range. To apply this operation, a random cut point is selected (uniform distribution) and from that point the segments from two chromosomes are interchanged.

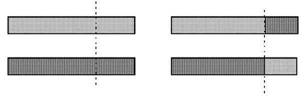

Figure 4. Crossover operation

Mutation (fig.5) plays a secondary role in GA, that's why the probability of applying it is less than 5% (generally speaking). This operation changes one bit (gene) of the chromosome. This change means to test a new occurrence that can be more suitable.

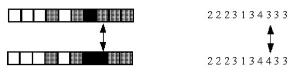

Figure 5. Mutation

In our experiment we finally accepted crossover with 70% probability of delivery, respectively mutation with 1% of progress (uniform distribution). The table rules was converted in a chromosome with $m * n$ gene (m, n are the numbers of membership functions for the two inputs). The gene was set to work in the numerical base p, where p is the number of membership function for the output of the controller.

For testing, the population size was set at 4, 10, and respectively 15 elements and the number of generations was 50 and 100. Best results was obtained for 10 elements of the population size and 50 generations of evolution algorithm. Best results were obtain for 10 elements of the population size, respectively 50 number of generations.

6 A hybrid GA–Trial-and-Error optimization. (rules optimization)

The genetic algorithms often provide an unsatisfactory result, as that shown in Fig.4 (output system evolution drawn with dashed line). On the other hand, the trial and error method becomes trapped in some cases (inability of adjustment; trapping in local minima). To cope with them, we apply the genetic algorithm technique to improve the system performance.

To correct these oscillations, the GA was followed by a second stage of adjustment, based on trial and error method. The continuos line represents the result got by using such a "cascade" of procedures, trial-and-error being applied more than once.

These frequent situations of oscillating output, can be interpreted as the system was not "soft" optimized (change the number of m.f., or the premise and conclusion are not well suited). It is known that increasing the number of m.f. the error is lowered (for suitable rules). Sometimes the slope of the output curve is good (the desired value is quickly reached in the first 400 steps), but the system is oscillating after that. One can conclude that we have to work on the set of rules that are often used in the corresponding interval. Thus, we have applied the trial and error method after the step number 400 — for this case. One has to repeat the trial and error procedure, for a m.f. number grater than 3. In this way one obtains a better solution [1,2].

The output simulated function evolution (the arterial blood pressure) is pictured below:

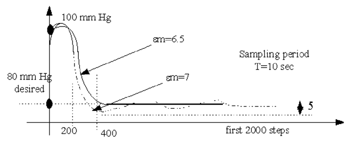

Figure 6. Blood pressure evolution

Below are presented some relevant results of simulation:

- the error is in the range: [-5,10]
- the change of error is in the range: [-0.5,0.5]
- the concentration of isoflurane: [0,4]

$$\mathit{total\ error} = \varepsilon_T = \sum_{i=1}^{n} |Y(i) - Y_{desired}|, \quad n = 2000 \tag{4}$$

$$\mathit{mean\ error} = \varepsilon_m = \frac{\varepsilon_r}{n} \tag{5}$$

$$\mathit{simulation\ function}: \ Y(n) = \sum_{i=1} ai \cdot Y(ni) + \sum_{j=1} bj \cdot X(mj) \tag{6}$$

Relevant results are presented in this table:

Table no.1.

mean error	error	change of error	fuzzy output	rules
	no. of m.f. / type			
7.75	2 Λ	2 Λ	2 Λ	1 1 2 2

mean error	error	change of error	fuzzy output	rules
7.75	2 Λ	2 Λ	3 Λ	1 2 3 3
8.05	2 Ω	2 Ω	3 Λ	1 2 3 3
	$\delta = 0.5$	$\delta = 0.5$		3 3
7.14	3 Λ	3 Λ	5 Λ	2 1 1 1 4 1 5 5 5
6.52	4 Λ	4 Λ	4 Λ	1 1 3 1 1 2 3 1 1 3 1 5 1 4 4 4
6.5	4 Λ	4 Λ	4 Λ	1 1 1 1 2 2 3 2 3 2 3 3 4 4 4 4
6.47	5 Λ	3 Λ	4 Λ	1 3 1 1 2 2 1 3 4 2 2 4 4 4 4

Λ represents the triangular type of m.f.; Ω indicates the Gaussian type of m.f. with δ standard deviation; mean error is computed for the first 2000 steps.

7 Software

The application runs on a 486 DX4 (minimum 4Mb RAM) computer using Visual Basic 3.0 software. The process functions, that can be simulated and controlled are the linear systems written in (7).

The user can define:

- The number of membership functions for inputs and outputs; – this number is limited by the computer memory.
- The membership functions type, that can by: triangular, trapezoidal or Gaussian. The base can be constant or can be multiplied by a constant greater than 0.
- The rules – table can be suggested. This is not mandatory: if the user does not provide the rules, the system proposes the m.f. number one, for the output m.f.

  The liner function that simulate the biological process:

$$Y(nT) = \sum_{i=1} ai \cdot Y(ni \cdot T) + \sum_{j=1} bj \cdot X(mj \cdot T) \qquad (7)$$

- The method for optimizing the set of rules:
  - * exhaustive combinatorial method – all possible combinations are tested
  - * trial and error
  - * statistical method (genetic algorithm)

The behavior of the system is represented with four pictures showing respectively: the simulated output evolution; the input simulated evolution; the phase diagram of the output; the evolution of error in time.

The type and the number of the membership functions can also be optimized by trial and error.

8 Results and discussion

First of all, the results consist in finding the set of rules that is optimum in most cases, for our application in an acceptable time (1–2 hours of calculus) for a big number of m.f. (4,5,7). Note in Fig.6 that the GA–TE optimization method leads to a control improved in all respects: the overshoot of the arterial pressure is lower and the error after 400 iterations becomes negligible. The final blood pressure after 400 calculus steps, has an error less than 1 mm Hg (desired set point 80mm Hg, start set point 100mm Hg).

The number and type of m.f. also influence the system performance. According to our simulations, it is obvious that for a given system there is a minimum number of m.f. to get an acceptable control [2]. Also the results give us the idea that a big part of knowledge is memorized in the system rules, because trying to optimize the type and parameters of the m.f. after the rules optimization there are not any changes in the m.f. type and parameters (in most of the experiences). There is a sense to increase the number of m.f. (to make the system more precise, soft), but there is a good reason in minimizing it, because a big number of m.f. implies a large computing time.

Also note to use an appropriate number of m.f., for example, in a symmetrical range value an odd number of m.f. is suited because the 0 value can be a stable state.

Below are pictured the results of our simulation. All pictures represent the mean blood pressure (mm Hg) evolution, on the first 1000 seconds.

Figure 7 represent the evolution of the simplified system with the function:

$$\begin{array}{l} y(k) = -a1 \cdot y(k-1) - a2 \cdot y(k-2) + b1 \cdot u(k-\tau1) + b2 \cdot u(k-\tau2) \\ a1 = -1.331, \quad a2 = 0.335, \quad b1 = -0.018, \quad b2 = -0.024 \\ \tau1 = 24[s], \quad \tau2 = 102[s] \end{array} \tag{8}$$

This system is similar with that described in [8] and represent a very good approximation.

Because of the biological diversity, the parameters $b1$ and $b2$ can be different from person to person. In paper [5] a similar study was done taking in to account the sensitivity of the patients.

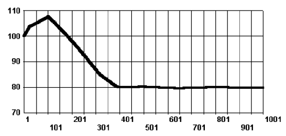

Figure 7. Blood pressure evolution for normal patients

For a sensitive patient the system was tested with the $b1 = -0.072$. The result is represented in the Figure 8.

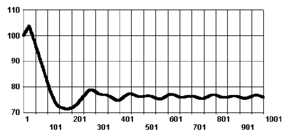

Figure 8. Blood pressure evolution for sensitive patients

The result is not satisfactory because of the overshoot of the system. In this situation it is necessary to improve the performances of the control. This type of control (adaptive control) will be developed in a new paper.

In Figure 9 the system was tested using the values $b1 = -0.014$ and $b2 = -0.02$. Again maintaining the same control parameters (op-

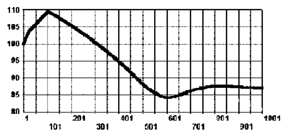

Figure 9. Blood pressure evolution for insensitive patients

timized for normal patients), the blood pressure evolution is not satisfactory.

Below in Figure 10, is presented the output evolution, for the system described by the function:

$$\begin{aligned} y(k) &= 1.2 \cdot y(k-1) - 0.18 \cdot y(k-2) \\ &\quad -0.1 \cdot u(k-5) - 0.007 \cdot u(k-7) \end{aligned} \tag{9}$$

This simple function was examined, to prove the control capacity of the system in the feedback configuration.

The control parameters are:

· number of m.f. for error: 3	range of working	-1. . . 5
· number of m.f. for change of error: 3	range of working	-0.5. . . 0.5
· number of m.f. for output control: 4	range of working	0. . . 3

· start set point 10 ; desired set point 5
· table of rules:

	change of error		
	2	1	1
error	4	4	4
	4	4	4

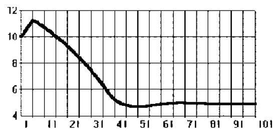

Figure 10. Output system evolution

9 Conclusions

The set of fuzzy rules was optimized using a cascade of two methods to get acceptable results. Namely, trial and error optimization after the genetic algorithm were used.

Indeed, trial and error algorithm does not ensure the earning of the global optimum. This method of finding the rules can successfully be used only after reaching a "good" starting set point, as well as the appropriate number of membership functions. It is necessary to go close to the global minimum error, for instance by using GA method.

To cope with the problem of local optima, we simulated the genetic algorithm that shift the system state closer to the global minimum error (in most cases). GA seldom gives the best solution. To improve the system performance we apply first GA and than "trial and error". By applying both methods, the global optimum is reached in most cases.

Trying to optimize the type and parameters of the m.f. we didn't find best results. We have explained this, by the knowledge that is implemented in a big part in the rules of the system. Because the system simulated function is linear, we do not expect any changes in the p-parameter of base m.f. variation.

References

[1] L.V.Boiculese, H.N.Teodorescu, G.Dimitriu. A Mixed G.A. -Trial-And-Error Optimization of Fuzzy Controllers. Medical Applications. International Conference on Intelligent Technologies in Human-Related Sciences, León, Spain, July 5–7, 1996.

[2] L.V.Boiculese, H.N.Teodorescu, G.Dimitriu. Optimizing Rules in Fuzzy Control for Anesthesia, Proceedings SCS, 1995, October 19–21 Iasi, Romania, p.101–104.

[3] Goldberg David E. Genetic Algorithms in Search, Optimization & Machine Learning, Adison – Wesley, 1989

[4] Guy Benchimol, Pierre Levine, Jean Charles Pomerol. Sisteme Expert In Intreprindere, Editura Tehnica Bucuresti, Ch.3, 1993

[5] Hao Ying, Michael McEachen, Donald W.Eddleman, Louis C.Shepperd. Fuzzy Control of Mean Arterial Pressure in Postsurgical Patients with Sodium Nitroprusside Infusion, 10, 1992, 1060–1069.

[6] Li Y., K.C.Tan, K.C. Ng and D.J.Murray-Smith. Performance Based Linear Control System Design by Genetic Evolution with Simulated Annealing,
http://www.elec.gla.ac.uk/reports/csc95017.htm

[7] E.W.McGookin, D.J.Murray-Smith and Y.Li. Segmented Simulated Annealing Applied to Sliding Mode Controller Design, submitted to: 13 Th. IFAC World Congress, San Francisco, CA, 1996

[8] R.Meier, J.Nieuwland, S.Hacisolihzade, D.Steck, A.Zbinden. Fuzzy Control of Blood Pressure During Anesthesia with Isoflurane, 2, 1992, 981–987.

[9] H.N.Teodorescu, Al.P.Tacu, J.Gil Aluja. Fuzzy Systems in Economy and Engineering. Publishing House of the Romanian Academy Ch.2, 1994, 3.

[10] H.N.Teodorescu, I.Bogdan, R.Strungaru. Fuzzy Systems and Neural Networks. Iasi, 1992, p.27–38.

[11] Witold Pedrycz. Fuzzy Control and Fuzzy Systems, Ch.2, 1989, 4.

[12] Yun Li and Kim Chwee NG. Genetic Algorithm Based Techniques for Design Automation of Three Term Fuzzy Systems. 1995 http://www.elec.gla.ac.uk/reports/csc95008.htm

L.V.Boiculese, G.Dimitriu,
University of Medicine and Pharmacy "Gr.T.Popa",
Department of Informatics,
16 Universitatii Street, Iasi,
6600 Romania,
phone: +40-32-114316,
e-mail: *boicules@umfiasi.ro*

Received 7 January, 1996

H.N.Teodorescu
Technical University "Gheorghe Asachi"
Department of Medical Electronics,
Bd.Copou no.11, Iasi,
6600 Romania,
phone: +40-32-142501,
e-mail: teo@tuiasi.ro