EVALUATION OF A MULTI-VARIABLE SELF-LEARNING FUZZY LOGIC CONTROLLER

Paul N. Ekemezie
Department of Electronic Engineering
University of Nigeria, Nsukka

ABSTRACT
In spite of the usefulness of fuzzy control, its main drawback comes from lack of a systematic control design methodology. The most challenging aspect of the design of a fuzzy logic controller is the elicitation of the control rules for its rule base. In this paper, a scheme capable of elicitation of acceptable rules for multivariable fuzzy logic controllers is derived by extending an algorithm that enables a single-input-single-output fuzzy logic controller to self-learn its rule-base. The performance of the proposed self-learning procedure is investigated and evaluated by means of simulation studies of a hypothetical plant. The results obtained indicate that the approach could be effective in the control of nonlinear multivariable industrial processes.

1. INTRODUCTION
For designing a classical or modern control system, it is necessary to have an accurate nominal model of the plant which is to be controlled, or the plant must have linearly parametrizable dynamics (such as in the case of adaptive control). In many engineering applications, however, this is impossible or very difficult to achieve due to complicated dynamics, severe nonlinearity and/or time-variance of the process. Modelling difficulties like these have forced control engineers to use simplified or linearised models, which are vulnerable to parameter inaccuracy (or structured uncertainty) and unmodelled dynamics (or unstructured uncertainty), with degraded control performance (Li et. al.,).

An approach adopted to combating (or limiting the effect of) these modelling difficulties is fuzzy logic control (FLC), which incorporates the uncertainty and abstract nature inherent in human decision making into expert-type control systems. These control systems produce control settings mainly based on the observed plant input/output behaviour (as opposed to a. model) and on the considerations of some characterisation of the model uncertainties, which have been found robust in implemented performance. However, fuzzy logic also has its down side. The major drawback of the engineering technology is the lack of a formal design methodology (Intel corporation, 1994; Lee et al., 2001; Hwang and Kuo, 2001). The other disadvantage is that the resulting system is not analytical, and therefore any mathematical analysis on paper is impossible with current methods.

The solution to the problem of construction of the rule base of a FLC has been the use of adaptive or self-learning techniques. Incidentally, most researchers have turned to the merger of fuzzy logic and other forms of soft computing (principally Neural Networks and Genetic Algorithms) in order to facilitate the self-tuning of control parameters. A FLC for a single-input-single-output (SISO) system that can self-construct

its rule base, and which does not need a merger of soft computing technologies, but instead is based on a purely fuzzy logic platform, was reported in (Ekemezie, 2000; Ekemezie and Osuagwu, 2001). Since most practical control systems are multivariable (MIMO), it is desirable to investigate the applicability of the reported self-learning FLC algorithm to a multivariable system, This is, the object of this paper.

Several techniques have been proposed that would enable a FLC to self- learn its rule base. The proposed techniques are either entirely based upon the corresponding process model identified in advance (Pedrycz, 1985; Procyck and Mamdani, 1979) or are learning-based, in which the rule base is gradually constructed by evaluating the control performance and providing a learning mechanism (Linkens and Nie, 1994). An indication that none of these methods has been completely satisfactory is the fact that researchers have resorted to marriage of fuzzy logic with other methods of Soft Computing, with the aim of achieving self-learning capability and computational efficiency. For example, Linkens and Nie (Linkens and Nie, 1994; Nie and Linkens, 1994a; Nie and Linkens, 1994b) proposed some self-learning FLC structures that assume availability of neither control experts nor teacher signals for the control problem. In each FLC structure they adopted an architecture that is similar to that of model reference adaptive control systems (MRAC). Each FLC structure was implemented as a neural network. They indicated that the advantages of the neural network implementation are computational efficiency and trainable capability of the network paradigm.

In formulating the self-learning FLC algorithm for the SISO system, some of the ideas developed by Linkens and Nie under neuro-fuzzy setting were adapted for use in a purely fuzzy logic platform.

The paper is organized as follows: section 2 reviews the fuzzy logic control system. The self-learning fuzzy logic control algorithm is described in section 3. Simulation studies carried out on the algorithm are presented in section 4. The paper ends with conclusive remarks.

2. FUZZY LOGIC CONTROL SYSTEM

The basic configuration of a fuzzy logic control system is shown in Fig. 1. The fuzzy logic control system performs a mapping from $U\hat{I}\hat{A}^n$ to $V\hat{I}\,\hat{A}$. Let $U = U_1 \times \ldots \times U_n$ where $Ui\hat{I}.\,\hat{A}i, = 1,2, \ldots . n$.

The fuzzifier maps a crisp point in U into a fuzzy set in U. The fuzzy rule base consists of a collection of fuzzy IF-THEN rules:

$$Rule\ k: IF\ \times_1 (t)\ is\ \Gamma_1^k\ and\ .\times_n (t) is\ \Gamma_1^k$$
$$THEN\ u(t)\ = \Omega^k$$

$$in\ which\ \times (t)$$
$$= (\times_1, \ldots . ,\times_n)^T \hat{I} U and\ u(t)$$
$$= (u_1(t), \ldots \ldots , U_m(t))^T \hat{I} V$$

are the input and output of the fuzzy system, $\Gamma_1^k and\ \Omega^k$ are fuzzy sets in U_i, and V, and k = 1,….,P where P denotes the number of fuzzy IF-THEN rules, The fuzzy inference engine performs a mapping from fuzzy sets in U to fuzzy sets in V, based upon the fuzzy IF-THEN rules in the fuzzy rule base and the compositional rule of inference. The defuzzifier maps a fuzzy set in V to a crisp point in V.

Let the membership function for every fuzzy set involved in the fuzzy logic control system be triangular, with the apex existing at the middle (or modal point) M of the support set and width a being the distance from the modal point to either of the left and right edges, as shown in Fig. 2:

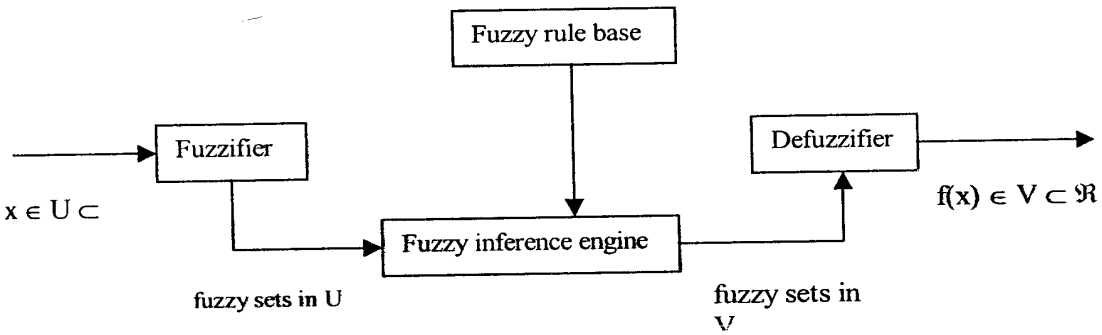

Fig. 1. Basic configuration of fuzzy logic control system (Hwang and Kuo, 2001).

The fuzzy set shown in Fig. 2 can also be regarded as a fuzzy number. Fuzzy numbers are fuzzy subsets of the real line. They have a peak or plateau with membership grade 1, over which the members of the universe are completely in the set. The membership function is increasing towards the peak and decreasing away from it. Fuzzy numbers are used very widely in fuzzy control applications. A typical case is the triangular fuzzy number (Fig. 2). In Fig. 2, the membership value at each point x in the region of the fuzzy number A is given by

$$\mu(\times) = 1 - \frac{|M - x|}{\alpha},$$
$$if\,|M - x| \le \alpha\ (2)0, if\,|M - x| > \alpha$$

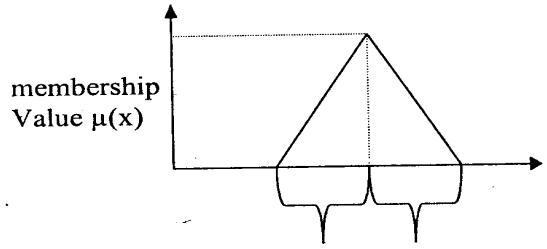

Fig. 2. Triangular membership function of a fuzzy set.

A triangular fuzzy number then can be characterized by only two items of data; M and α With the use of triangular fuzzy numbers as the fuzzy sets in Eq. (1), the inference procedure needs to be modified since the desired result is no longer a set of output fuzzy sets in V, but a set of strings of fuzzy numbers in V. The inference procedure involves obtaining every M inferred by each triggered rule and associating them with the activation strength (or truth value) of the rule. Hence, the goal of the Inference engine is to obtain pairs of the form {M, u}, where M is an output fuzzy number and μ represents the strength of the fuzzy number's contribution to the eventual crisp solution.

A defuzzification method needs to be chosen that will suit the result from the inference procedure. The weighted averaging method of defuzzification (Nie and Linkens, 1994b) is particularly suitable. In this method, the jth control output is given by

$$uj = \frac{\sum_{q-1}^{Q} \mu^q M_u^q}{\sum_{q-1}^{Q} \mu^q} \qquad (3)$$

In equation (3), Q rules have been triggered in the inference process; μ_q is the activation strength of the rule and M_u^q is the centre value of an output fuzzy number

3. The self-learning algorithm

Fig. 3 shows the overall structure of the self-learning FLC. It consists of a conventional FLC plus a reference system and an adaptation mechanism. The fuzzy reasoner is given an initial rule-base containing what may be an arbitrary set of rules at the design stage. The function of the adaptation mechanism is to iteratively modify the contents of the rule-base so as to attain the desired control objectives. The desired performance of the control system is designated by the reference model, which

specifies what the process responses should be when both of them are subject to the same command signal (Linkens and Nie, 1994).

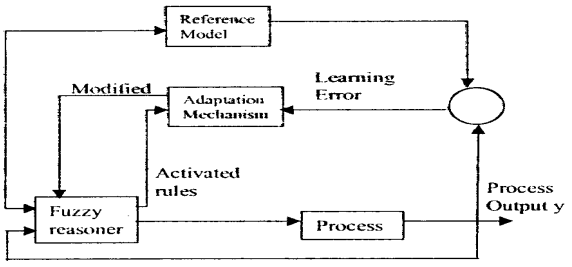

Fig. 3. Overall structure of the self-learning FLC

3.1 The Reference Model

For a multivariable process, the following non-interacting model with second-order linear transfer functions are adequate:

$$H(s) = diag\{H_1(s), H_2(s), \ldots\ldots, H_n(s)\} \quad (4)$$

Where H(s) is a diagonal transfer matrix relating the reference signal r to the model output y_r, and $H_i(s)$ is given by

$$H(s) = \frac{\omega_n^{2} e^{-\tau s}}{S^2+2\xi\omega_n S+\omega_n^2} \quad (5)$$

where ω_n is the undamped natural frequency, ξ is the damping ratio of the system and τ is the delay time. The models given above are linear, but do not suggest that the process itself must be linear. Yet they do suggest that some knowledge about the process should be available. With some prior knowledge of the process and well-known linear control theory, it is not difficult to determine the parameters in the above models by specifying the desired time domain indices or by allocating the pole positions in the s-plane (Linkens and Nie, 1994).

3.2 The adaptation mechanism

Both the closed-loop control system and the reference mode are given the same set point. During each sampling period k the adaptation mechanism compares the process output y_p and reference model output y_r as follows:

$$e_L(k) = y_r(k) - y_p(k) \quad (6)$$

Where e_L is the leaning error.

If the process response exhibits a delay time estimated to be γ sampling periods, in each sampling period the adaptation mechanism recalculates the centroid for every rule that was triggered γ sampling periods before the present sampling period, but makes use of the current leaning error. The updating of each rule's centroid is governed by the following equation:

$$M_U(K) = M_U(K+\gamma) + \rho.^{e_L}(k).\mu, \quad (7)$$

Where M_u (k) is the new centroid, μ is the activation strength of the rule and p is the leaning rate. In order to implement Eq. (7), the adaptation mechanism will need to store an array of past triggered rules obtained from the fuzzy reasoner, with their associated predicate truth values.

4. SIMULATION STUDIES

4.1. The Control system

The objective of the simulation studies is to investigate the performance of the self-leaning algorithm by observing the response of the control system. The controlled plant is assumed to be described by the transfer matrix:

$$T(S) = \begin{bmatrix} \frac{1}{(S+1)^2} & \frac{1}{(S+1)(S+2)} \\ \frac{1}{(S+1)(S+2)} & \frac{S+3}{(S+2)^2} \end{bmatrix} \quad (8)$$

Eq. (8) represents a system with two outputs, y_1 and y_2 Two control inputs, u_1 and u_2, are required. This means that there will be two error signals $e_1$and e_2, from which will be calculated two change-in-error signals, c_1,and c_2. Each input is decomposed into seven fuzzy regions, whose centroids and span (a) are as follows.

$$m_i = \{-1, -2/3, -1/3, 0, 1/3, 2/3, 1\}\alpha = 1/3\} \quad (9)$$

In the course of experimentation, it was discovered that the self-learning scheme achieved its objective only when the FLC

design assumes that the process is completely decoupled, that is, $_{U1}$ affects only y_1 and u_2 only y_2. Consequently, the required controller is basically two single-input-single-output FLCs operating in parallel. Each FLC requires a maximum of 7^2 rules. Initially, the rule-base is filled with all zeros, that is, the centroid of the consequent fuzzy number for each rule is made equal to zero.

The reference model is in the form of Eq. (4), but having only H_1 and H_2 Both $H_1$and H_2 are in the form of Eq. (5), with $\omega_n = 3, \xi = 0.75$ and τ = O. The adaptation mechanism used a learning rate $\rho = .05$.

4.2. Results and discussion

Fig. 4 shows the output responses of the process in comparison with the reference model after the first simulation cycle. Fig. 5 shows the output responses of the process in comparison with the reference model after the twentieth simulation cycle. Fig. 6 shows the output responses of the process in comparison with the reference model after the fiftieth simulation cycle.

It can be noticed that as the number of cycle's increases, the output of the process approaches the reference model output. It can also be noticed that, after the 50th cycle, y_2 followed the reference model better than y_1.*This* is because the response of y_1, is slower in comparison with the speed of reference model. The implication of this is that the choice of a reference model for a given variable should be done realistically.

As stated is § 4.1, it was discovered that the self-learning scheme achieved its objective only when the FLC design assumes that the process is completely decoupled. This makes it to be consistent with the approach of industrial process control (Cao et al. 1998). In industrial process control, a complex control system is decomposed into several control loops, and each control loop uses a simple control law. The whole control system is then a combination of these simple control loops. Since process variables are usually nonlinear, to find a set of local stabilization control actions is much easier than to find a global stabilization control action for the whole system.

5. Conclusion

A scheme capable of automatic elicitation of suitable rules for a multivariable fuzzy logic controller has been proposed. The scheme requires some modifications of the conventional fuzzy system model. The modifications include the use of fuzzy numbers as input and output of the fuzzy reasoner and the employment of the weighted averaging method of defuzzification. Also the rules are made to specify the centroid of a fuzzy number, instead of a linguistic value as their consequents.

The performance of the proposed self-learning procedure has been investigated by means of simulation studies of a hypothetical plant. The results obtained from the simulation studies indicate that the self- learning procedure actually adapts the rule- base of an FLC to a given process.

In the course of experimentation, it was discovered that the self-learning scheme achieved its objective only when the FLC design assumes that the process is completely decoupled. This has been shown to be consistent with the approach of industrial process control, in which a complex control system is decomposed into several control loops, and each control loop uses a simple control law. Therefore, the proposed control scheme is amenable to industrial applications.

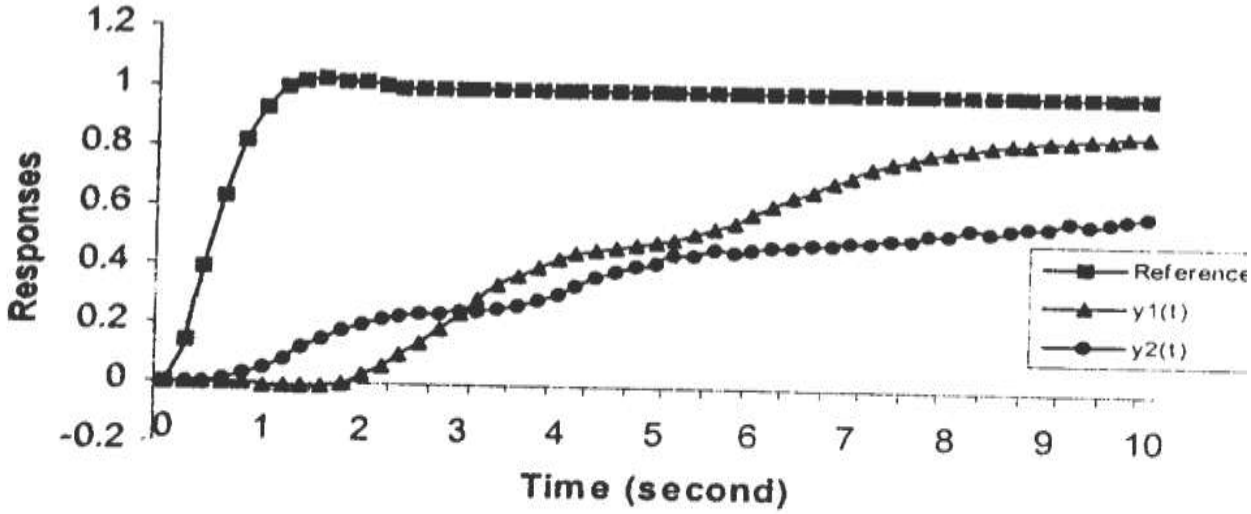

Fig. 4. Responses after the 1st simulation cycle.

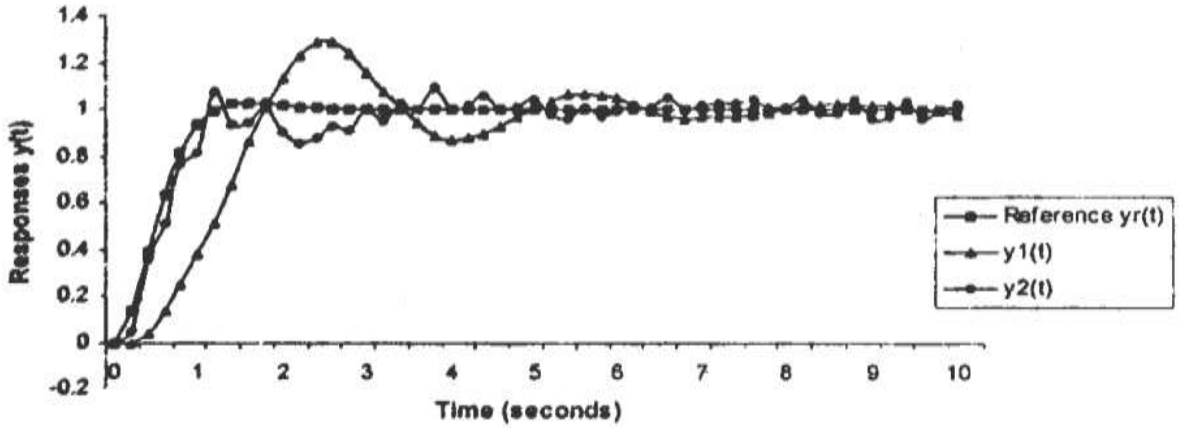

Fig. 5. Responses after the 20th simulation cycle.

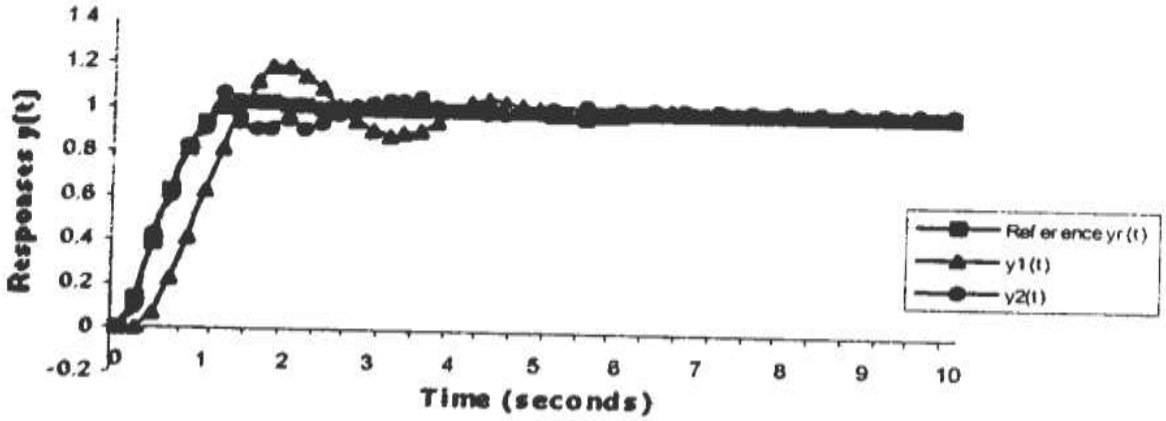

Fig. 6. Responses after the 50th simulation cycle.

REFERENCES

Cao, S.G., Rees, N.W. and Feng, G., 1998. "Lyapunov-like stability theorems for continuous-time fuzzy control systems. *Int. J. Control*, Vol. 69, NO.1, pp. 49-64.

Ekemezie, P. N., 2000, "*Investigation of the influence of design parameters on the performance of fuzzy logic controllers and formulation of a self-organising control strategy*." Ph.D Thesis, University of Nigeria, Nsukka.

Ekemezie, P. N. and Osuagwu, C. C. 2001, "A self-organising fuzzy logic controller." *Nigerian Journal of Technology*, Vol. 20, No. 1, March 2001, pp. 1-8.

Hwang, C.L. and Kuo, C.Y., 2001, "A stable adaptive fuzzy sliding-mode control for affine nonlinear systems with application to four-bar linkage systems." *IEEE Trans. Fuzzy Systems*, Vol. 9, No.2, pp. 238-252.

Intel Corporation, 1994, MCS® 1996 Microcontrollers -- The Perfect Match for Fuzzy Logic Applications." http://www.intel.com/design/mcs96/papers/

Lee, H.J., Park, J.B. and Chen., G., 2001, "Robust fuzzy control of nonlinear systems with parametric uncertainties IEEE *Trans. Fuzzy Systems*, vol, 9, No.2,pp.369-379.

Linkens, D. A. and Nie, J., 1994, "Back-propagation neural-network based fuzzy controller with a self-learning teacher." *Int. J. Control*. Vol. 60, No.1, pp 17-39.

Nie.J. and Linkens, D. A., 1994, "A hybrid neural-network-based self-organizing controller." *Int. J. Control.* vol. 60, no. 2, pp 197-222.

Nie.J. and Linkens, D. A., 1994, "Fast self-leaning multivariable fuzzy controllers constructed from a modified CPN network." *Int. J. Control*. vol. 60, no. 3, pp 369-393.

Pedrycz, W.,1985."Design of fuzzy control algorithms with the aid of fuzzy models," in Industrial Applications of Fuzzy control, ed. by M. Sugeno. North Holland, Amsterdam.

Procyck, T. and Mamdani, E. H., 1979, "A linguistic self-organising controller" *Automatica*, vol. 15, pp. 15-30.

Takagi T. and Sugeno M., 1985. "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Sys. Man and Cyber.* Vol. SMC-15, pp. 116-132.