Fuzzy Reasoning Implemented by Neural Networks

Junhong Nie and D. A. Linkens

Department of Automatic Control & Systems Engineering
University of Sheffield, Sheffield S1 3JD, U.K.

Abstract: Viewing the given rule-base as defining a global linguistic association constrained by fuzzy sets, approximate reasoning is implemented here by a Backpropagation Neural Network (BNN) with the aid of the fuzzy ste theory. By paying particular attention to the capability of generalization of the BNN, the underlying principles have been examined in detail using two examples. The simulation results not only indicate the feasibility of the BNN-based approach, but also reveal some deeper similarities which exist in the two methods, which may have some important implications for future studies of fuzzy control.

1. Introduction

Fuzzy or approximate reasoning (AR) employed by rule-based fuzzy expert systems or fuzzy logic controllers can be regarded as a process by which a set of imprecise conclusions is deduced from a collection of imprecise premises [1]. Although the reasoning can be carried out by fuzzy logic-based reasoning algorithms, here we consider another possibility of implementation of AR by neural networks. The motivation comes mainly from the fact that the goals of AR and mapping neural networks are very similar, that is, to perform some kind of approximation or interpolation, although the representation the information they are dealing with is different. In particular, we claim that neural networks inherently possess some fuzziness, the source of approximation in AR, which is displayed in the network via net-computing in the form of generalization. This view is of particular importance in the net-based fuzzy control as will be discussed in section 5.

Instead of seeking a structure mapping from a fuzzy reasoning system to a neural network, this paper is intended to find a functional mapping from the fuzzy logic-based algorithm to the network-based approach. By viewing the given rule-base as defining a global linguistic association constrained by fuzzy sets, the approximate reasoning is implemented by Backpropagation Neural Networks (BNN) with the aid of fuzzy ste theory. By paying particular attention to the capability of generalization of the BNN, the underlying principleshave been examined in detail using two examples, a small demonstration at the linguistic level, and a more realistic problem of multivariable fuzzy control of blood pressure. The simulation results not only indicate the feasibility of the BNN-based approach, but also reveal some deeper similarities which exist in the two methods, which may have some important implications for future studies of fuzzy control. In addition, this work may be considered as another appliction example of the BNN in the case of continuous ouputs and on a relatively larger scale (in the second example,the BNN has 26 inputs and 16 outputs, with a total of 2013 weights and thresholds). Furthermore, the work may provide evidence to support the argument that net-computing is, in fact, knowledge representation as claimed very recently by Pao [2].

2. Formulation of the problem

Assume that the system has n inputs and m outputs denoted by $X_1, X_2, \cdots X_n$ and $Y_1, Y_2, \cdots, Y_m$. Furthermore, it is assumed that L "IF *situation* THEN *action* " rules, and n inputs in the IF part and m outputs in the THEN part are connected by linguistic connectives ALSO and AND respectively. Then the problem may be described as follows:

- Given rules: $RULE^1$ ALSO $RULE^2$ ALSO ⋯⋯ ALSO $RULE^L$
  where $RULE^j$ has the form:
  IF X_1 is A_1^j AND X_2 is A_2^j AND ⋯⋯ AND X_n is A_n^j
  THEN Y_1 is B_1^j AND Y_2 is A_2^j AND ⋯⋯ AND Y_m is A_m^j , j=1,2, ⋯⋯, L.
- Given input data: X_1 is C_1 AND X_2 is C_2 AND ⋯⋯ AND X_n is C_n
- To find output data: Y_1 is D_1 AND Y_2 is D_2 AND ⋯⋯ AND Y_m is D_m
  where X_i and Y_k are linguistic variables whose values are taken from the discourses of universe U_i, V_k with $U_i=(u_{i1}, u_{i2}, \cdots\cdots, u_{is_i})$ and $V_k=(v_{k1}, v_{k2}, \cdots\cdots, v_{kr_k})$. A_i^j, C_i, B_k^j, D_k are fuzzy subsets which are defined on the corresponding discourses, and represent some fuzzy concepts such as *big, medium* and *small* etc. More precisely, fuzzy subsets A_i^j, C_i, B_k^j, D_k are characterized by the corresponding membership functions $A_i^j(u_i) : U_i \to [0, 1]$, $C_i(u_i): U_i \to [0, 1]$, $B_k^j(v_k): V_k \to [0,1]$,

0-7803-0559-0 /92 $3.00 © 1992 IEEE

and $D_k(v_k) : V_k \rightarrow [0,1]$.

The above problem can be solved using fuzzy-logic-based algorithms in the following two ways. The first one may be called the relational matrix method. The basic idea is first to create a relational matrix R based on the given L rules using some logical operators. Then the outputs for a specific input is determined by Zadeh's compositonal rule of inference. The other solution is a straightforward one in the sense that the algorithms closely follow the inferencing procedures used by data-driven reasoning systems and which take the fuzzy variables into account [3]. The basic idea is first to treat the L rules one-by-one and then to combine the individual results to give a global output.

3. Solutions using neural networks

Although the Backpropagation Neural Network (BNN), the Basis Function Network (BFN), and Probability Function Network (PFN) are good candidates for our purpose, the well-known BNN [4] structrure has been chosen for the current study. In what follows , we will formulate the BNN and AR from the functional viewpoint so as to obtain a formal equivalence between them.

From the mapping perspective, the goal of the BNN is to perform an approximate implementation of an unkown mapping ϕ, from a compact set $I^{N_I} \subset R^{N_I}$, an N_I demensional Euclidean space, to an N_O demensional Euclidean space, ϕ: $I^{N_I} \rightarrow R^{N_O}$, by an approximator ϕ^* consisting of layered and massively connected processing units. Assume that the topology of the network is specified. Then constructing ϕ^* is equivalent to determining the parameters of the BNN based on a set of selected examples. More specifically, assume that we are given a set of examples (u^1, v^1), $\cdots$, (u^P, v^P) with u^p being drawn from I^{N_I}, and v^p being supposed to be satisfied with the unknown function ϕ, i.e. $v^p = \phi(u^p)$. Denote all the weights and thresholds in the structure-specified BNN as a parameter vector w. Furthermore, the P desired and actual outputs calaculated from the BNN at P discrete sample points are denoted as $\phi \equiv [\phi^1(u^1), \phi^2(u^2), \cdots, \phi^P(u^P)]$ and $\phi^* \equiv [\phi^{1*}(u^1, \mathrm{w}), \phi^{2*}(u^2, \mathrm{w}), \cdots, \phi^{P*}(u^P, \mathrm{w})]$ respectively. Then the problem can be simply formulated as to select w in such a way that a specific error fuction, say, a quadratic one, as denoted by

$$E(w) = \frac{1}{2} || \phi(u) - \phi^*(u,w) ||^2 \tag{1}$$

is minimized, provided that the strucure of the BNN is specified *a priori*. We can expect with some confidence that the approximator constructed from only a finite set of points in I^{N_I} will work satisfactorily over the whole space of I^{N_I}.

Based on the idea discused above, the problem of the approximate reasoning presented in section two may be solved if it is formulated in the same way as in the BNN, that is, to construct an inference engine Ψ^* based on the given L rules and to generalize to unseen situations in the domain of interest by directly manipulating the engine. Clearly, a given rule is analogus to an example in the BNN. L IF-THEN statements may designate an implicit and global relationship between the situation set and action set. To be more specific, denote the situation variable X=[$X_1, X_2, ..., X_n$] and the action variable Y =[$Y_1, Y_2, ..., Y_m$]. X and Y take linguistic labels, represented by fuzzy sets A_i^j and B_k^j defined on the corresponding universes U_i and V_k, as their values, for example, X=X^p=[$A_1^p, A_2^p, ..., A_n^p$]. Further, let Ψ(X)=[$Y^1, Y^2, ..., Y^P$] and Ψ^*(X, W)=[$\Psi^*(X^1,W), \Psi^*(X^2,W), ..., \Psi^*(X^P,W)$] be the desired action and the actual action calculated from the inference engine Ψ^* with respect to P situations X^p, p=1,P, where W denotes a set of parameters determining the Ψ^*. Then, by analogy to equ 1, the problem may be reformulated to select W in such a way that

$$\tilde{E}(W) = \frac{1}{2} || \Psi(X) - \Psi^*(X,W) ||^2 \tag{2}$$

is minimized such that an action Y=[$D_1, D_2, ..., D_m$] will be approximately deduced from the constructed inference engine Ψ^* when a new situation X=[$C_1, C_2, ..., C_n$] is encountered. Thus, the Ψ^* accomplishes an approximate implimentation of linguistic mapping from one linguistic set of the situation domain to another linguistic set of the action domain.

It is obvious, from the above discussions, that the AR problem can be solved by the network method if the linguistic values can be translated into numerical forms suitable for the use of the BNN and the numerical outputs from the BNN can be interpreted as liguistic labels. The graded membership function is the first choice for the former transformation and some methods for linguistic approximation may be used for the latter conversion. It is noted that although the reasoning system itself has n iuputs and m outputs, the BNN using the fuzzy set representation will have $s_1+s_2+ \cdots +s_n$inputs and $r_1+r_2+ \cdots +r_m$ outputs with

s_i and r_k being the cardinality of U_i and V_k on which fuzzy sets are defined. There exist other possibilities for the label/numerical value transformation, one of which will be discussed in section 5.

4. Reasoning capability: a linguistic study

The main objective of the study in this section is two-fold, i.e. to investigate the capability of the BNN-based reasoning system in the linguistic level and to examine the effect of the number of hidden units upon this capability, by means of the simulation on a small but typical problem.

Suppose we are given five rules, each of which has the form of "IF X is A THEN Y is B" with the pair (A, B) being specified as (PB, NB), (PM, NM), (ZR, ZR), (NM, PM) and (NB, PB), where PB, PM, ZR, NM, and NP stand for Positive-Big, Positive-Medium, Zero, Negative-Medium and Negative-Big respectively. For simplicity, we assume that all the fuzzy sets representing the above linguistic labels are defined on the same discourses of universe, i.e. U=V={-4,-3,-2,-1,0,1,2,3,4,} and all the membership functions are of triangular form located at -4, -2, 0, 2, 4. The BNN consists of one input and one output layer with 9 input and 9 output units, corresponding to the cardinalities of U and V, and one hiden layer with a variable number of units. By setting the initial weights to be uniformly distributed on [-0.5, 0.5], the net was trained by the presentation of five rules with the learning rate being 0.8 and the number of hidden units being 5, 15, 30 and 50 respectively. The training process was stopped when the sum of squared error within one cycle was less than 0.0005. Then the BNN was tested using the following two sets of linguistic labels.

The first set of linguistic inputs used so-called linguistic hedges to modify the basic labels in the rules. Here two hedges, *very* and *more or less* were used and defined as *very A* $\equiv A^2$ and *more or less A* $\equiv A^{0.5}$. Thus, ten different inputs could be used to test the performance. For example, if *very-positive-medium* is presented to the BNN, the expected outputs should be *very-negative-medium*

Instead of merely altering the shapes of the basic labels as above, the second set of linguistic inputs was concerned with changing the central values with respect to the basic labels, representing a much harder situation than the previous one since the inputs are substantially different from that in the given rules. It is convenient to interpret the linguistic labels as fuzzy numbers meaning linguistically that " X is about u ". With the same trained BNN, Fig. 2 gives the results corresponding to the expected outputs of "about" +3, +1, -1, -3. At first sight, one may think that the results are not as good as would be expected. However, if we calculate the area under the different curves and accept this quantity as a global measure of the performance, the results are remarkably good. For example, when the expected output is "about -1", the absolute differences between the expected and actual areas (calculated using a weighted sum) are 0.1552, 0.1141, 0.1142, and 0.0534 corresponding to 5, 15, 30, and 50 hidden units respectively.

It should be noted that the generalization performance of the BNN does not monotonously increase with an increase in the number of the hidden units. To clarify this point, we adopt another global measure, namely the centre of gravity (COG) which is an important quantity in fuzzy control, where it is used to produce a non-fuzzy output. For the present problem, the averaged differences between the expected and actual COG were 0.1239, 0.0912, 0.1109, and 0.0965 corresponding to 5, 15,30, 50 hidden units respectively, indicating that, at least in the current context, the reasoning performance could be degraded with too few or too many hidden units.

5. Reasoning capability: a fuzzy control example

This section presents a more complicated and realistic problem solved by the BNN-based reasoning method. The problem is to regulate simultaneously the cardiac output (CO) and mean arterial pressure (MAP) of a patient in hospital intensive care using an inotropic drug dopamine (DOP) and a vasoactive drug sodium nitroprusside (SNP). The simulation model including the cardiovascular system and drug dynamics is given by

$$\begin{bmatrix}\Delta CO\\ \Delta MAP\end{bmatrix}=\begin{bmatrix}1.0 & -24.76\\ 0.6636 & 76.38\end{bmatrix}\begin{bmatrix}\dfrac{K_{11}e^{-\tau_1 s}}{sT_1+1} & \dfrac{K_{12}e^{-\tau_2 s}}{sT_1+1}\\ \dfrac{K_{21}e^{-\tau_2 s}}{sT_2+1} & \dfrac{K_{22}e^{-\tau_2 s}}{sT_2+1}\end{bmatrix}\begin{bmatrix}\Delta DOP\\ \Delta SNP\end{bmatrix} \tag{3}$$

This problem has been investigated by the authors [3] using the logic-based fuzzy control approach. For the purpose of comparisons, we employ the same architecture as used in [3] except that here the fuzzy controller is implemented using the BNN-based method. The control system, as shown in Fig.3, consists of two separated control loops, CO/DOP and MAP/SNP, with the aid of a simple compensator to reduce the interactive effects. For simplicity, the two controllers, each comprising two linguistic inputs, error e and change-in-error ce, and one liguistic output u, are assumed to be identical and therefore only one of them is described.

Suppose that all the discourses of unverse have the same form consisting of 13 integers ranged from -6 to 0 to 6, and that seven linguistic labels (negative-big (medium, small), zero, and positive_big (medium, small)) are used. They are defined by the same trangular form with the central values located at -6,-4,-2,0,2,4,6 respectively. Thus the resultant BNN has 26 inputs and 13 outputs. If one hidden layer with 50 units is used, the BNN will have a total of 2013 adjustable parameters.

Two phases, off-line training and on-line application, are needed. Using the given 33 rules as shown in Table 1, the BNN was trained with 50 hidden units and a learning rate of 0.02. Although the actual outputs of the BNN are in the interval [0,1] in accordence with the range of a sigmoid nonlinear active function, it has been found that a faster convergency could be achieved if linear active functions in the output layer are used, as illustrated in Fig.4. To investigate the effects of the number of rules upon the reasoning performance, a set of 16 rules selected from 33 rules as marked with "*" in Table 1 was also trained. As expected the sum of the squared error decreases more quickly as shown in Fig.4.

Because the measured error e, change-in-error ce and the required control u are numerical, the e and ce have to be fuzzified into fuzzy sets and the fuzzy sets output of the BNN must be defuzzified into a real number during the application stage. Here singleton fuzzification and COG defuzzification methods are employed. However, if the linguistic labels are viewed as fuzzy numbers and the BNN is trained with the central values of the fuzzy numbers only, the complexity of the controller is greatly reduced and the efficiency in speed and storage is dramaticaly increased due to the fact that the number of inputs and output is reduced to 2 and 1 and neither fuzzification nor defuzzification is needed. To verify this possibility, the BNN with 15 hidden units was trained and tested using 33 and 16 rules respectively. The results are reported next.

After appropriate training, the BNN was used as the controller in the system. The capability of the generalization of the BNN was evaluated indirectly by the control performances measured by two frequently used indices: integral of square of the error (ISE) and integral of time and absolute error product (ITAE) for each output. We considered four cases: the BNN's with linear active functions in the output layer trained by fuzzy sets with 33 and 16 rules (denoted by FS_33 and FS_16) and by fuzzy numbers with 33 and 16 rules (FN_33 and FN_16). Table 2 shows the simulation results when the system was subject to square-like inputs as shown in Fig. 6. The results obtained using the logical-based method [3] are also included in Table 2 for comparison purpose. Due to space limitation, only the output responses in the case of FS-33 are presented in Fig.5.

The following conclusions can be drawn from the results: (a) the BNN possesses a high ability to generalize, a very useful property for fuzzy reasoning. This is particularly demonstrated by the case of FS_33 and FS_16 where the measured singleton inputs were very different from the training inputs;(b) it is possible to use fewer rules to obtain a valid generalization as illustrated by FS_16 and FN_16; (c) perhaps the most important conclusion from the results is that the BNN trained by the fuzzy numbers can produce performances as good as, or even better than, the one trained by the fuzzy sets as indicated by the case of FS_33 wrt. FN_33 or FS_16 wrt. FN_16; (d) the similar performances obtained by the logic-based and the BNN-based reasoning approaches suggest the feasibility of the BNN for the application of fuzzy control.

6. Conclusion

By paying particular attention to the capability of generalization of the BNN, it has been demonstrated that a forward-chaining fuzzy reasoning system with parallel rule-bases can be implemented within the framework of neural networks. The studies into the BNN-based fuzzy controller suggest that, besides a seeming resemblence between rules and patterns in the logic-based and BNN-based approaches, there exists a deeper similarity in the information processing aspect in them, namely, fuzziness vs. distributiveness. The authors believe that this suggestion has some important implications for the development of fuzzy control under the framework of neural networks. Some studies along these lines are currently in progress.

REFERENCES:

[1]. L.A. Zadeh, Outline of a new approach to the analysis of of complex systems and decision processes, *IEEE Trans. Syst. Man Cybern.*, 3 , pp 28-44, 1973.

[2]. Y.-H. Pao & D. J. Sobajic, Neural networks and knowledge engineering, *IEEE Trans. Knowledge and Data Engineering*, **3** , pp 185-192, 1991.

[3]. D. A. Linkens & Junhong Nie, A unified real time approximate reasoning approach for use in intelligent control: Part 1 & 2, submitted, 1991.

[4]. D. Rumelhart, G. Hinton and R. Williams, Learning internal representation by error propagation. in *Parallel Distributed Processing* , Vol. 1, Rumelhart & McClelabd Eds., MIT press,1986.

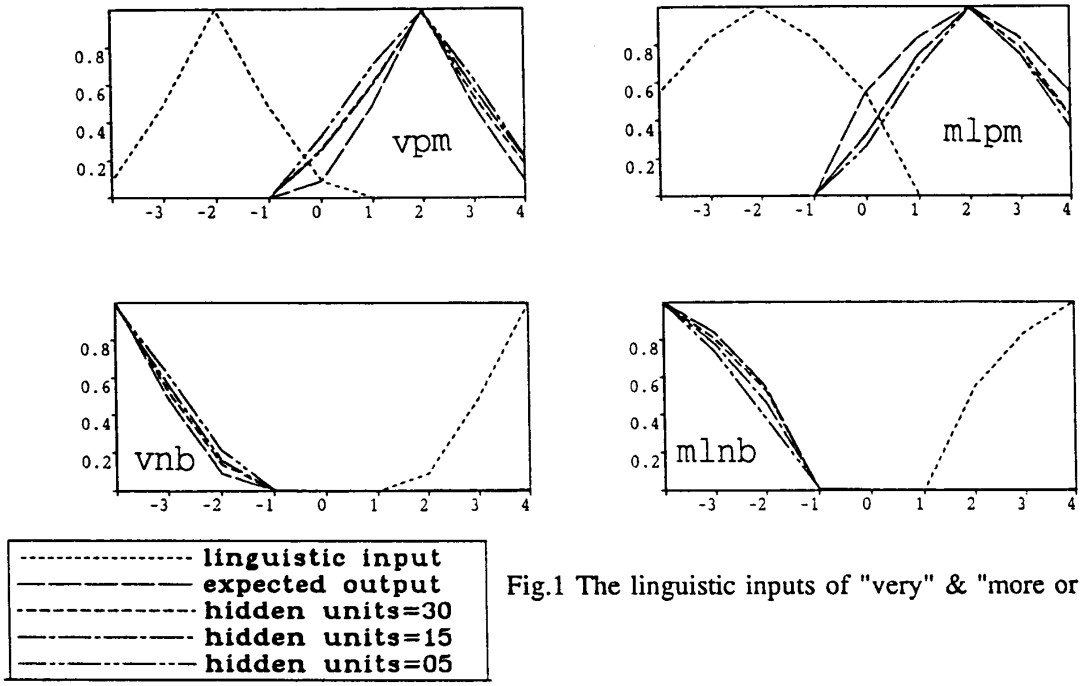
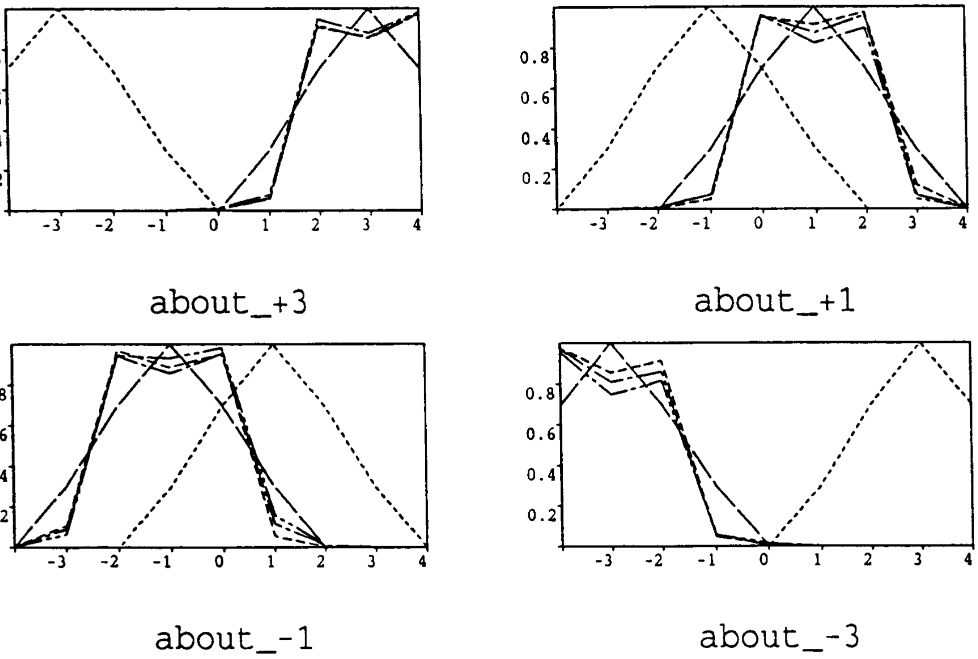


Fig.1 The linguistic inputs of "very" & "more or less"



about_+3

about_+1

about_-1

about_-3

Fig.2 The linguistic inputs of "about"

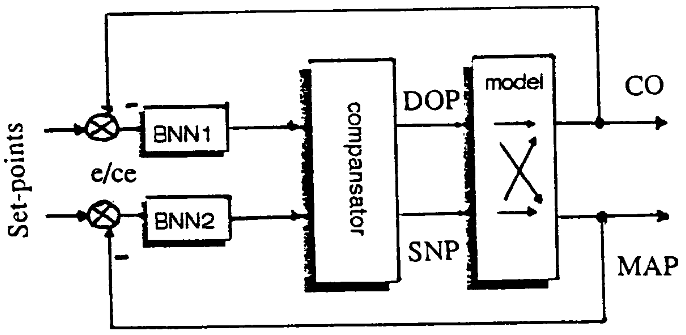
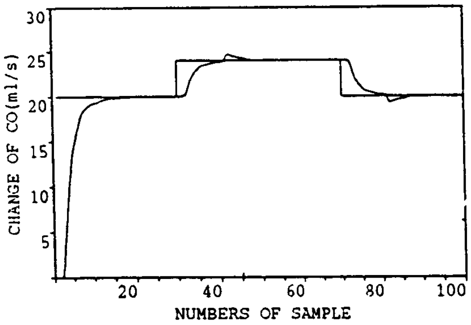


Fig.3 BNN-based fuzzy control system



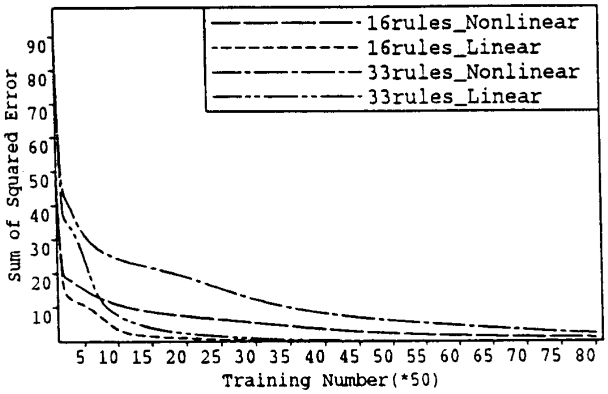


Fig.4 SSE vs. training cycles

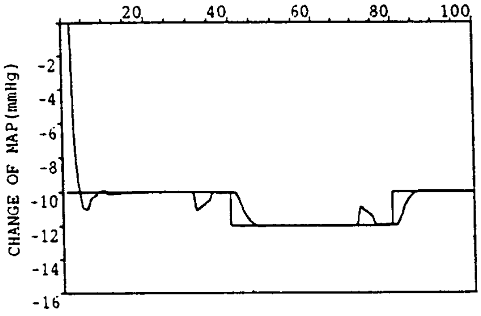


Fig.5 Output responses of the system with net of FS_33

Table 1 Control rules

E \ C	NB	NM	NS	ZR	PS	PM	PB
NB		NB		NM	NM	NS	
NM	NB		NM		NS		PS
NS		NM *	NS	NS *	ZR *	PS *	PM
ZR	NB		NS *	ZR *	PS		PB
PS	NM	NS *	ZR *	PS *	PS *	PM *	
PM	NS *		PS *		PM *		PB
PB		PS *	PM	PM *		PB	

*: 16 rules

Table 2 Control Performances

	CISE	CITAE	PISE	PITAE
FS_33	16.80	843.47	3.55	586.24
FS_16	16.83	902.16	3.54	609.35
FN_33	16.79	810.29	3.49	560.72
FN_16	16.78	810.27	3.50	558.87
MMC	16.77	808.43	3.48	557.06

* C for CO and P for MAP