An Adaptive Control Using Fuzzy Logic Neural Network and Its Application*

WANG Yaonan

(Department of Computer, East China Geological Institute • Jiangxi Fuzhou, 344000, PRC)

Abstract: This paper presents an adaptive neural fuzzy intelligent control scheme. The proposed neural fuzzy adaptive control consists of the fuzzy Gaussion neural network and model neural network, and has two important characteristics of adaptation and learning. The effectiveness of the proposed scheme will be demonstrated by computer simulations and the practical application in the servo motor control.

Key words: fuzzy logic; neural network; adaptive control; servo control

1 Introduction

For many control processes it is difficult to obtain a precise mathematical model of the system. This may happen if the system is very complex, nonlinear, uncetain variations of the environments including the load disturbance, and the sudden change of the plant parameters, etc. In order to overcome these difficulties, recently, a considerable number of researches on the intelligent control system with a human-like inference and adaptation ability have been proposed[1,2,3]. This paper proposes an adaptive neural fuzzy intelligent control scheme to combine the neural network with fuzzy logic. The proposed control scheme consists of the neural fuzzy controller (FGNC) and model neural network (MNN) as shown in Fig. 1. In the FGNC, the fuzzy antecedent processing, fuzzy inferencing, and defuzzification are constructed by a multi-layer Gaussion neural network. The FGNC is trained to refine the fuzzy rules and turn the membership functions of fuzzy reasoning for the adaptively inferenced control action by the error - back propagation (BP) learning algorithm[2]. The

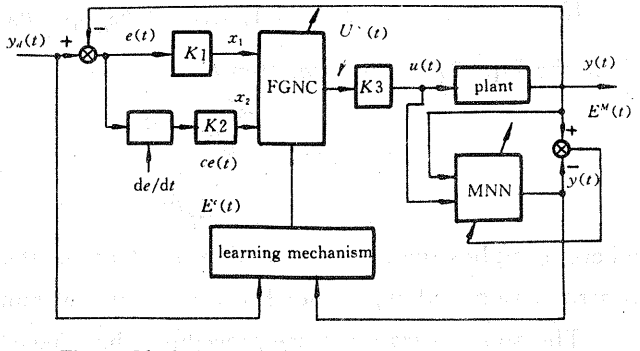

Fig. 1 Block diagram of adaptive neural fuzzy control

* This research is supported by Sate Key Pattern Recognition Laboratory Foundation, Chinese Academy of Science.

Manuscrpt received Jun. 7, 1994, revised Nov. 19, 1994.

MNN is constructed to identify an unknown controlled plant and provied the FGNC with the teaching signal. The MNN is continuously trained on-line by the BP learning algorithm. We used the proposed adaptive control scheme to the servo motor control and obtained encouraging results.

2 Neural Fuzzy Controller

2.1 Fuzzy Logic Control

In general, the dynamic behavior of a fuzzy logical controller is characterized by a set of linguistic control rules based on the knowledge of an expert. Let's assume that the linguistic control rules can be expressed as follows[3].

$$\begin{aligned} R_i: \quad & \text{if } (x_1 \text{ is } A_{i1}) \text{ and } (x_2 \text{ is } A_{i2}) \text{ and } \cdots \text{ and } (x_n \text{ is } A_{in}) \\ & \text{then } (U_1 = B_{i1}) \text{ and } \cdots \text{ and } U_p = B_{ip} \end{aligned} \tag{1}$$

where R_i denotes the ith control rule, $x_j (j = 1,2,3,\cdots,n)$ and $U_i (i = 1,2,3,\cdots p)$ are the input and control variables, respectively; A_{ij} is the fuzzy variable characterized by fuzzy membership function $\mu_{A_{ij}(x_j)}$; B_{ij} is real constant value.

In this paper the simplified reasoning methods regarded as a special case of product-sum method will be utilized. The reasoning process can be expressed as follows

$$U_j^* = \frac{\sum_{i=1}^{m} (\mu_{A_{i1}(x_1)} * \mu_{A_{i2}(x_2)} * \cdots * \mu_{A_{in}(x_n)}) \cdot B_{ij}}{\sum_{i=1}^{m} (\mu_{A_{i1}(x_1)} * \mu_{A_{i2}(x_2)} * \cdots * \mu_{A_{in}(x_n)})}, \qquad j = 1,2,\cdots,p, \tag{2}$$

where the symbol " * " denote a algebraic product.

If

$$\mu_i = \prod_{j=1}^{n} \mu_{A_{ji}(x_j)} = \mu_{A_{i1}(x_1)} * \mu_{A_{i2}(x_2)} * \mu_{A_{in}(x_n)} \tag{3}$$

then, the Eq. (2) can be rewritten as follows:

$$U_j^* = \frac{\sum_{i=1}^{m} \mu_i \cdot B_{ij}}{\sum_{i=1}^{m} \mu_i}, \qquad j = 1,2,\cdots,p. \tag{4}$$

where μ_i implies the truth value of antecedence of the ith rule corresponding to the fuzzy control action, and B_{ij} are not fuzzy sets but real numbers.

The above fuzzy inference procedure, the antecedent and consequent parts of the "if-then" rule can be constructed by a multilayer Gaussian neural networks with nonlinearity and learning function. Fig. 2 shows the proposed structure of the FGNC.

2.1 Neural Fuzzy Controller

Fig. 2 shows the implementation of the fuzzy logic control using the FGNC.

Let's assume that each control rule has two input variables (x_1, x_2) and one output variable (u^*). It is also assumed that each input variable has seven membership functions. The term set of each fuzzy variable is {NB, NM, NS, ZO, PS, PM, PB}, where NB, NM, ···, are abbreviations for the commonly used names "Negative Big", "Negative

Medium", and so on.

The layers (Ⅰ), (Ⅱ) in Fig. 2 correspond to the antecedent part of the fuzzy control rules, and the layers (Ⅲ)～(Ⅳ) correspond to the conclusion part. The input-output relationships of units in the FGNC are defined as

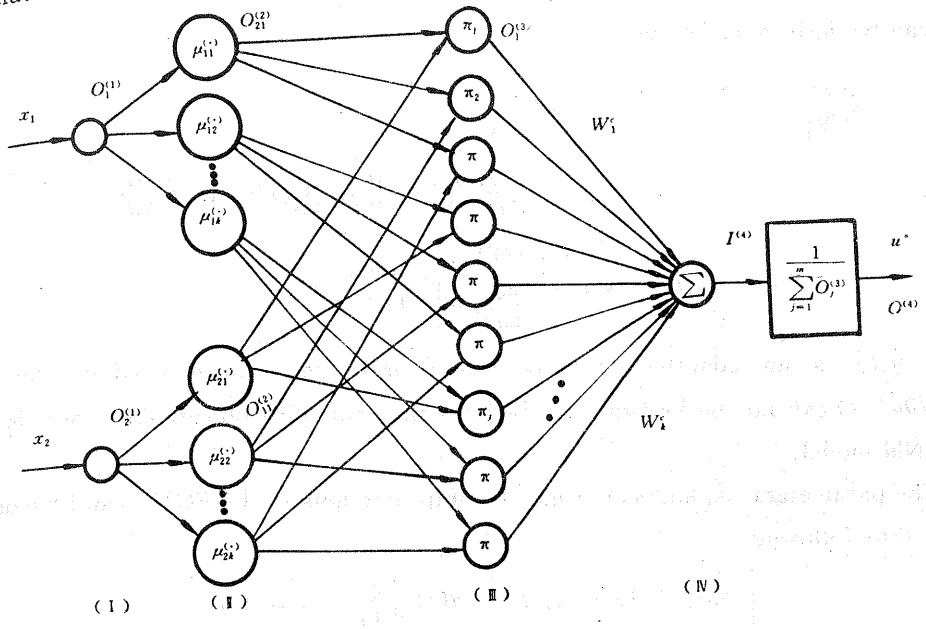

Fig. 2 A structure of neural fuzzy control

(Ⅰ) input units: $O_i^{(1)} = x_i, \quad i = 1,2;$ (5)

(Ⅱ) input units: $I_{ik}^{(2)} = -(x_i - a_{ik})^2/b_{ik};$ (6)

output units: $O_{ik}^{(2)} = \mu_{ik} = \exp(I_{ik}^{(2)}), \quad i = 1,2; k = 1,2,\cdots,n;$ (7)

(Ⅲ) input units: $I_k^{(3)} = \prod_{i=1}^{2} O_{ik}^{(2)}$, output units: $O_j^{(3)} = I_j^{(3)}, \quad j = 1,2,\cdots,n^2.$ (8)

Where $O_j^{(3)}$ is the truth value of the jth fuzzy rule. $\prod$ is the product operation of the grades of the membership function ($\mu_{ik}(\cdot)$).

(Ⅳ) input units: $I_k^{(4)} = \sum_{j=1}^{m} O_j^{(3)} \cdot W_j^c,$

output units: $O^{(4)} = u^* = \dfrac{I^{(4)}}{\sum_{j=1}^{m} O_j^{(3)}}, \quad j = 1,2,\cdots,m.$ (9)

After constructing the antecedent parts and the conclusion part of fuzzy rule by a neural network, next step is to train the connection weights of the FGNC, and to identify the control rules and to tune the membership functions parameters (a_i, b_i) in the antecedent part. In order to obtain the inference control in the FGNC, BP algorithm is used for learning to minimize difference E^c between the desired output $y_d(t)$ and the actual output $y(t)$ of the plant. An error function to be minimized is defined as follows:

$$E^c = (1/2) * (y_d(t) - y(t))^2, \tag{10}$$

then, the weight W_j^c in the FGNC is changed as

$$W_j^c(t+1) = W_j^c(t) - \eta(t)\frac{\partial E^c}{\partial W_j^c} + \beta\Delta W_j^c(t). \tag{11}$$

$\frac{\partial E^c}{\partial W_j^c}$ can be derived as follows

$$\begin{aligned}\frac{\partial E^c}{\partial W_j^c} &= \frac{\partial E^c}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial u^*(t)} \cdot \frac{\partial u^*(t)}{\partial W_j^c} \\ &= -(y_d(t) - y(t))\left(\frac{\partial y(t)}{\partial u^*(t)}\right) \cdot \frac{\partial}{\partial W_j^c}\left(\sum_{j=1}^m O_j^{(3)} \cdot W_j^c \Big/ \sum_{j=1}^m O_j^{(3)}\right) \\ &= -(y_d(t) - y(t))\left(\frac{O_j^{(3)}}{\sum_{j=1}^m O_j^{(3)}}\right)\left(\frac{\partial y(t)}{\partial u^*(t)}\right).\end{aligned} \tag{12}$$

where $\eta(t)$ is an adaptive learning rate, β is a momentum constant, the term $\partial y(t)/\partial u^*(t)$ can not be known directly from the controlled object. This value is from the MNN model.

The parameters (a_{ik} and b_{ik}) of membership functions in the FGNC can be modified or tuned as follows:

$$\begin{cases} a_{ik}(t+1) = a_{ik}(t) - \eta(t)\dfrac{\partial E^c}{\partial a_{ik}} + \beta\Delta a_{ik}(t), \\ b_{ik}(t+1) = b_{ik}(t) - \eta(t)\dfrac{\partial E^c}{\partial b_{ik}} + \beta\Delta b_{ik}(t). \end{cases} \tag{13}$$

$\frac{\partial E^c}{\partial a_{ik}}$ can be derived as follows:

$$\begin{aligned}\frac{\partial E^c}{\partial a_{ik}} &= \frac{\partial E^c}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial u^*} \cdot \frac{\partial u^*}{\partial a_{ik}} \\ &= -(y_d(t) - y(t))\left(\frac{\partial y(t)}{\partial u^*}\right)\left[\frac{W_k^c \cdot \sum_{j=1}^m O_j^{(3)} - \left(\sum_{j=1}^m O_j^{(3)} \cdot W_j^c\right)}{\left(\sum_{j=1}^m O_j^{(3)}\right)^2}\right] \cdot 2(x_i - a_{ik}) \cdot O_k^{(3)}/b_{ik} \\ &= -(y_d(t) - y(t))\left[W_k^c \sum_{j=1}^m O_j^{(3)} - \left(\sum_{j=1}^m O_j^{(3)} \cdot W_j^c\right)\right] \\ &\quad \cdot 2(x_i - a_{ik}) \cdot O_k^{(3)} \Big/ \left[b_{ik}\left(\sum_{j=1}^m O_j^{(3)}\right)^2\right] \cdot \left(\frac{\partial y}{\partial u^*}\right).\end{aligned} \tag{14}$$

$\frac{\partial E^c}{\partial b_{ik}}$ can be derived as follows:

$$\begin{aligned}\frac{\partial E^c}{\partial b_{ik}} &= \frac{\partial E^c}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial u^*} \cdot \frac{\partial u^*}{\partial b_{ik}} \\ &= -(y_d(t) - y(t))\left(\frac{\partial y(t)}{\partial u^*}\right) \cdot \frac{\partial u^*}{\partial O_k^{(3)}} \cdot \frac{\partial O_k^{(3)}}{\partial b_{ik}}\end{aligned}$$

$$= -(y_d(t) - y(t))\left(\frac{\partial y(t)}{\partial u^*}\right)\left[\frac{W_k^c \cdot \sum_{j=1}^{m} O_j^{(3)} - \left(\sum_{j=1}^{m} O_j^{(3)} \cdot W_j^c\right)}{\left(\sum_{j=1}^{m} O_j^{(3)}\right)^2}\right] \cdot O_k^{(3)}(x_i - a_{ik})^2/b_{ik}^2$$

$$= -(y_d(t) - y(t))\left[W_k^c \cdot \sum_{j=1}^{m} O_j^{(3)} - \left(\sum_{j=1}^{m} O_j^{(3)} \cdot W_j^c\right)\right]$$

$$\cdot (x_i - a_{ik})^2 \cdot O_k^{(3)} \Big/ \left[b_{ik}^2\left(\sum_{j=1}^{m} O_j^{(3)}\right)^2\right] \cdot \left(\frac{\partial y(t)}{\partial u^*}\right). \tag{15}$$

3 Adaptive Neural-Fuzzy Control

In order to identify the dynamics of the plant and obtain the Jacobian $\left(\frac{\partial y(t)}{\partial u^*}\right)$ of Eqs. (12), (14), (15), in this section, a learning scheme using the model neural network (as shown in Fig. 3) is used for system identification.

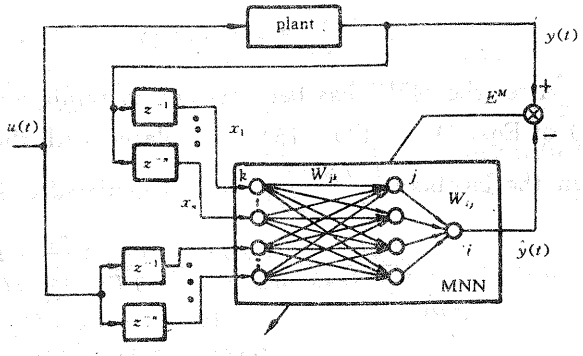

Fig. 3 Block diagram of system identification using the MNN

In Fig. 3 the input of the MNN are the current states $X_k(t)$, $(k = 1, 2, \cdots)$ of the plant and the control input $u(t)$ corresponding to the scaled output of the FGNG. And the output of the MNN is the next states $X_k(t+1)$ of the plant. The input-output relationship of the MNN are defined as follows:

$$\text{Input units:} \quad X_k^{(1)}(t) = \begin{cases} y(t-k), & 1 \leqslant k \leqslant n, \\ u(t-k+n), & n+1 \leqslant k \leqslant n+m, \end{cases} \tag{16}$$

where $y(t) = f(y(t-1), \cdots, y(t-n), u(t-1), \cdots, u(t-m))$.

$$\text{Hidden units:} \quad O_j^{(2)} = f(I_j^{(2)}), \quad j = 1, 2, \cdots, m_2, \tag{17}$$

$$I_j^{(2)} = \sum_{k=1}^{m_1} W_{jk} \cdot X_k^{(1)}, \quad k = 1, 2, \cdots, m_1. \tag{18}$$

$$\text{Output units:} \quad \hat{y}(t) = f(I_i^{(3)}), \tag{19}$$

$$I_i^{(3)} = \sum_{j=1}^{m_2} W_{ij} \cdot O_j^{(2)} \tag{20}$$

where W_{jk} is the connection weight from the kth input unit to the jth hidden unit, W_{ij} is the connection weight from the jth hidden unit to the ith output unit, $f(I) = 1/(1 + \exp(-I))$, $\hat{y}$ is the output of the MNN, $O_j^{(2)}$ is the output from the jth hidden unit. The error function used for training the MNN is defined as follows:

$$E^m = (1/2) \cdot \sum_{i=1}^{n} (y_i(t) - \hat{y}_i(t))^2, \tag{21}$$

where $y_i(t)$ is the output of the plant.

In order to minimize the error function E^m, the weight value, (W_{ij}, W_{jk}) of the MNN can be updated by the BP algorithm as fallows:

$$W_{ij}(t+1) = W_{ij}(t) - \eta(t)\frac{\partial E^m}{\partial W_{ij}} + \alpha(W_{ij}(t) - W_{ij}(t-1)), \tag{22}$$

$$W_{jk}(t+1) = W_{jk}(t) - \eta(t)\frac{\partial E^m}{\partial W_{jk}} + \alpha(W_{jk}(t) - W_{jk}(t-1)), \tag{23}$$

where

$$\frac{\partial E^m}{\partial W_{ij}} = \frac{\partial E^m}{\partial \hat{y}(t)} \cdot \frac{\partial \hat{y}(t)}{\partial I_i^{(3)}} \cdot \frac{\partial I_i^{(3)}}{\partial W_{ij}}$$

$$= -(y_i(t) - \hat{y}_i(t))\hat{y}_i(t)(1 - \hat{y}(t))O_j^{(2)} = -\delta_i \cdot O_j^{(2)}, \tag{24}$$

$$\frac{\partial E^m}{\partial W_{jk}} = \frac{\partial E^m}{\partial I_j^{(2)}} \cdot \frac{\partial I_j^{(2)}}{\partial W_{jk}} = \sum_i \left(\frac{\partial E^m}{\partial I_i^{(3)}} \cdot \frac{\partial I_i^{(3)}}{\partial O_j^{(2)}}\right) \frac{\partial O_j^{(2)}}{\partial I_i^{(2)}} \cdot X_k$$

$$= -\sum_{i=1}^{m_2} \delta_i \cdot W_{ij} O_j^{(2)}(1 - O_j^{(2)}) \cdot X_k = -\delta_j \cdot X_k. \tag{25}$$

After the MNN has been trained to emulate exactly like a plant, the plant output $y(t)$ of Eqs. (12), (14), (15) are replaced with the MNN output $\hat{y}(t)$ $(y(t) \approx \hat{y}(t))$. Then, the Jacobian $\left(\frac{\partial y(t)}{\partial u^*(t)}\right)$ can be calculated as follows:

$$\frac{\partial y(t)}{\partial u^*(t)} = \frac{\partial \hat{y}(t)}{\partial u^*(t)} = \frac{\partial \hat{y}(t)}{\partial I_i^{(3)}} \sum_{j=1}^{m_2} \frac{\partial I_i^{(3)}}{\partial O_j^{(2)}} \cdot \frac{\partial O_j^{(2)}}{\partial I_j^{(2)}} \cdot \frac{\partial I_j^{(2)}}{\partial X_k^{(1)}}$$

$$= \hat{y}(t)(1 - \hat{y}(t)) \sum_{j=1}^{m_2} W_{ij} \cdot O_j^{(2)}(1 - O_j^{(2)}) W_{jk},$$

$$(n + 1 \leqslant k \leqslant m + n). \tag{26}$$

To increase the convergence speed of the learning algorithm, we employ an adaptive learning rate $\eta(t)$. After starting with a small learning rate, its modifications are descsribed by the iterative equation

$$\eta(t) = \begin{cases} d\eta(t-1), & \text{if } E(W(t)) < E(W(t-1)), \\ e\eta(t-1), & \text{if } E(W(t)) \geqslant kE(W(t-1)), \\ \eta(t-1), & \text{otherwise} \end{cases}$$

where typical value of the parameters are $d = 1.05, e = 0.7, k = 1.04$.

4 System Simulation and Application

We assumed that the physical parameters of the plant are completely unkown. We used the 49 fuzzy control rules as shown in Table 1. The constant values within this table are set as the initial connection weight: $W_j^c(0)$ for each FGNC. The center points of the fuzzy sets {NB, NM, NS, ZO, PS, PM, PB}, $a_{ik}(0)$ $(k = 1, 2, \cdots, 7)$ are $-6, -4, -2, 0, 2, 4, 6$, respectively. The width values of the membership functions, $b_{ik}(0)$ are all unity so as to equally allocate seven fuzzy sets on the range $[-6, 6]$ $(b_{ik}(0) = 1.5^2)$.

4.1 System Simulations

In system simulaton, the FGNC is constructed by the 2—14—49—1 neuron, and the

corresponding MNN is constructed by the 4—10—1 neuron, the learning rates $\eta(0) = 0.25, \alpha = 0.2, \beta = 0.2$ respectively. The transfer function of the servo motor system is simplified to $G(s) = \dfrac{0.58}{s(0.4s+1)(0.05s+1)}$.

Table 1 fuzzy control rules

(x_1) / e_i	NB	NM	c_{ei} NS	(x_2) ZO	PS	PM	PB
NB	− 6.0	− 6.0	− 4.0	− 6.0	− 4.0	− 4.0	− 4.0
NM	− 6.0	− 4.0	− 2.0	− 4.0	− 4.0	− 4.0	2.0
NS	− 4.0	− 2.0	− 2.0	− 2.0	0.0	− 2.0	4.0
ZO	− 4.0	− 4.0	− 2.0	0.0	2.0	4.0	6.0
PS	− 4.0	− 2.0	0.0	2.0	2.0	2.0	4.0
PM	2.0	4.0	4.0	4.0	2.0	4.0	6.0
PB	4.0	4.0	4.0	6.0	4.0	6.0	6.0

where, the servo amplifier has an output range of ± 10V, $e(t) = \pm 6$V, $ce(t) = \pm 30$V, The output error $e(t)$, error change $ce(t)$, and control input variable are quantified to range of $[-6, 6]$. The scaling factors are $k_1 = 6/e(t) = 1.0, k_2 = 6/ce(t) = 0.2, k_3 = 10/6 = 1.82$, respectively.

Fig. 4 shows the system simulation results that the learning of the FGNC is made. Fig. 5 shows the simulation results of the conventional fuzzy control +PID.

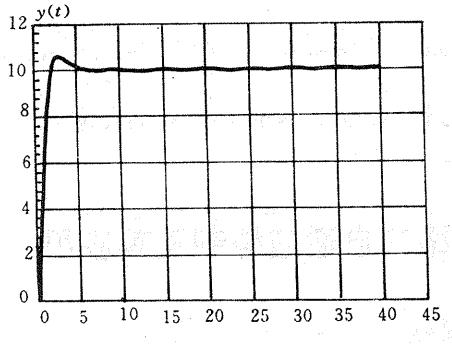

Fig. 4 Output response of the FGNC control system

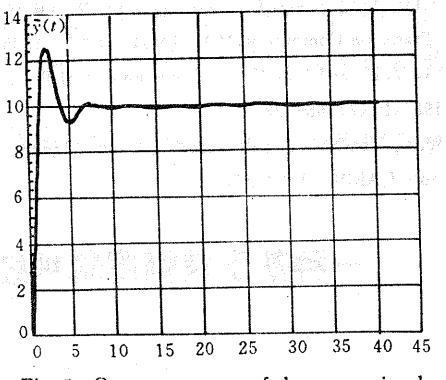

Fig. 5 Output response of the conventional fuzzy control+PID

4.2 Application to a Servo Motor Control

After system simulations procedure, the proposed adaptive control scheme is applied to the real-time control of the servo DC motor system. The struture of the servo DC motor system composed of control computer, interface and servo motor plant (1.5kW, 220V, 1500 rpm) is shown in Fig. 6. Sampling time interval $t = 2.5$ms. The step output response of the servo motor

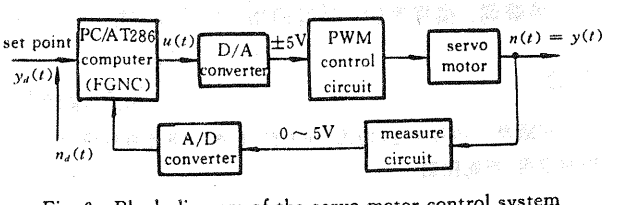

Fig. 6 Block diagram of the servo motor control system

system is shown in Fig. 7, where a time response with fast rise time and small overshoots is achieved.

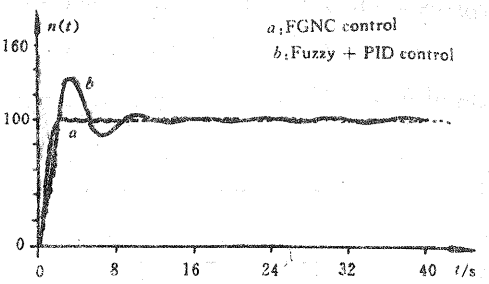

Fig. 7 Output response of the servo motor

5 Conclusion

The simulation results and the practical application of the servo system show that the proposed neural fuzzy control is of better performance than the conventional fuzzy controller for the servo system, so that the servo system has fast response and minimum steady-state error. The proposed control scheme can be also used in the unknown dynamics of the plant and the control of the nonlinear plant.

References

[1] Chung-chien. Lee. A Self-Learning Rule-Based Controller Employing Approximate Reasoning and Neural Net Concepts. Int. J. Intelligent System, 1991, 16(1): 71—93

[2] Rumelhart, D. and Mcclelland, J.. Parallel Distributed Processing. Cambridge, MA: MIT Press, 1986

[3] Lee. C. C.. Fuzzy logic in Control System: Fuzzy Logic Controller-Part Ⅰ, Ⅱ. IEEE Trans. Syst. Man, Cybern, 1990. 20(2): 408—435

[4] Chin Tenglin and C. S. Geroge Lee. Neural Network Based Fuzzy Logic Control and Decision System. IEEE Trans. on Computers, 1991, 40(12): 1320—1326

[5] Li, Y. F. and Lau. C. C.. Development of Fuzzy Algoritms for Servo Systems. IEEE Control System Magazine, 1989, 21(2): 65—92

[6] Wang Yao Nan. A Self-Learning Control System Using Neural Networks. In Proc. Modelling, Simulation & Control of AMSE, 1992, 2(1): 344—350

一种基于模糊逻辑神经网络的自适应控制及其应用

王耀南

(华东地质学院计算机系·江西抚州, 344000)

摘要：本文提出了一种模糊逻辑神经网络自适应控制器. 这种控制器由一个模糊高斯神经网络和一个多层神经网络组成. 它具有自适应和学习能力. 计算机仿真和实际的伺服直流电机调速实验的结果表明本文提出的这种控制器是切实可行的, 其系统响应和鲁棒性优于常规的 Fuzzy 控制.

关键词：模糊逻辑；神经网络；自适应控制；伺服控制

本文作者简介

王耀南 1958年生. 博士, 副教授. 主要从事计算机应用、工业自动化、人工智能及专家系统、神经网络理论及控制中的应用、智能控制.