広島大学総合科学部紀要Ⅳ理系編，第21巻，81－91，1995年12月

ニューラルネットワークによるファジィ制御器の学習について

水上孝一*・普家浩文**

*広島大学総合科学部、大学院工学研究科情報工学専攻
**㈱ILC（前工学研究科情報工学専攻大学院博士課程前期）

On Learning of Fuzzy Contoller with Neural Network

Koichi MIZUKAMI* and Hirofumi FUKE**

*Faculty of Integrated Arts and Sciences, Graduate school of Engineering (Information Engineering), Hiroshima University, Higashi-Hiroshima 739, Japan
**ILC, Inc (Former, Master Student of Information Engineering, Graduate school of Engineering, Hiroshima University, Higashi-Hiroshima 739, Japan)

Abstract : The fuzzy controller (FC) consists of two parts. First one is the control rule part which is referred to as linguistic rules. And it is written in the form of "IF ～ THEN ～ ". Second one is fuzzy reasoning which derives the reasoning value from the control rules. So it is easy for us to understand FC. However, there are two problems. First one is how to make the control rules. We usually take the rules from an expert. But it is not easy. Because all rules which the extpert has are not linguistic rules.

The neural network has ability of learning. So many people expect that it is useful to get the control rules identified from the expert. And several researchers study with respect to it. Horikawa, one of them, presents a fuzzy controller using a neural network. The controller can automaticaly identify the control rules and tune the membership functions utilizing control data of the expert. The controller uses the rules written in the form of "IF ～ THEN y = f", where 'f' is constant. However, when FC uses this rules, it needs many rules.

In this paper, we present a new fuzzy controller with a neural network which uses the rules written "IF ～ THEN y = f(・)", where f(・) is a linear function. Because when FC uses this rule, it need not many rules.

In order to demonstrate the effectiveness of this FC, we simulate by using a computer to control a first order delayed system with a dead time. And we compare this controller with Horikawa's controller.

Key words : Neural Network, Fuzzy Controller, Learning, Control Rules

1995年8月31日受付；1995年10月12日受理

1. はじめに

ファジィ制御は、制御規則と、その規則から操作量を決定する推論部から構成され、制御規則は普通 IF ～ THEN 形式の言語的表現を用いて記述される。そのため、人間にとり非常に分かりやすい制御器の構成が可能であり、様々な分野で実用化が行われている。しかし、文献[1]においてファジィ制御に関する問題点として次の２つを指摘している。

1．人間（エキスパート）は、制御規則を必ずしも IF ～ THEN 形式の言語的表現で持っているわけではなく、ある種の「感」のようなものが存在することも多々ある。そういった制御規則を IF ～ THEN 形式で構築するには、エキスパートからの聞き取り調査や、その後の制御規則の調整には多大な苦労が伴う。
2．実際の制御対象では、環境や動作条件の変化によってその動特性が変化したり、またその動特性が未知な場合もある。このような状況下にあってある特定の制御対象に対する制御規則しか持たないファジィ制御では、そのような制御対象を制御することは困難となる。

これらの問題に対して近年、前者に対してその解決策として様々な研究が進められている[2)－5)]。一方、後者に対しては、前者に比べれば少ないものの、その研究もまた進められている。

本論文では、前者の問題を取り上げ、その解決法としてニューラルネットワークを利用した方法を通じてその議論を進めてゆく。

ニューラルネットワークは、生態系の模倣であり、学習能力が優れている。そのため、前者のようなファジィ制御器の問題点を、その学習能力によって解決することが有効であると期待されている。そこで、ファジィ制御とニューラルネットワークの技術を融合するには、それぞれの類似点を接点とする方法と、異なる特徴を組み合わせる方法がある。

これらの方法の一方法として文献[5]が挙げられる。この文献では、ファジィ制御におけるメンバシップ関数を、ニューラルネットワークにおけるシグモイド関数で表現し、さらにニューラルネットワークをファジィ推論が行われるように構成することにより、学習型ファジィ制御器を設計している。この方法で使用されているファジィ推論法は、制御ルールの後件部を定数[6]で表現し、前件部適合度を乗算で、推論値を前件部適合度と後件部定数との積和で求めている。この推論法の特徴として

1．このファジィ推論法は、Mamdani の提案した後件部をメンバシップ関数（ファジィ数）で表現した推論法を簡略化したものであり、非ファジィ化におけるファジィ数の面積計算をする必要がない。
2．Mamdani の推論法と同様に、言語的に表現されている人間（エキスパート）の経験的な制御規則を記述する上で直感的に理解しやすい。
3．Mamdani の推論法と実用的に大差がない。

などが挙げられ、ファジィ制御の研究や実際の適用例において最も多く利用されている推論法である。

以上の議論はファジィ制御に関した問題であるが、文献５）に見られるようなこの種の研究はこ

の機能だけでなく、システム同定などのファジィモデルなどの他の機能を有している。上述のような簡略化された推論法は、単に制御規則を記述する（モデリング）という点においてその有用であるが、同時に次のような問題を持っている。

1．後件部を定数にすることにより、制御規則の記述に多くのルール数を必要とする。
2．直感的に理解しやすい反面、より深い解釈ができない。

そこで本論文では、制御ルールの後件部を線形式[7]で表現するファジィ推論法により制御部を設計する方法を提案する。この方法の特徴として

1．ルール記述能力に優れ、少ないルール数で制御器を構成可能である。
2．局所的な線形関係で表現されているため、階層型ニューラルネットワークモデルや多項式モデルなどの非線形モデルよりもシステムの動特性の性質を理解しやすい。
3．この推論法は後件部の線形式に定数項を含むため、後件部を定数にした推論法を包含している。

などが挙げられ、本論文の方法は、制御器の構成だけにとどまらず他のファジィモデルの構成にも有効である。

本論文の第2章では、提案するニューラルネットワークによるファジィ制御器の構成法と、学習法について述べる。第3章では簡単なシミュレーションにより本制御器と文献4)における制御器との比較をおこう。第4章はまとめである。

2．ニューラルネットワークによるファジィ制御器

2．1　ファジィ推論法

ファジィ制御におけるファジィ推論法は、数多くのものが提案されている[5),6)]。文献4)で使用されるファジィ推論法は、次のようなものである。

2入力（x_1, x_2）1出力（y^*）の制御器の場合、ルールおよび推論法は以下のように表記される。

制御器への入力が（x_1^0, x_2^0）の場合、次のように表される。

$$R^i : \text{IF } x_1 \text{ is } A_{i1} \text{ and } x_2 \text{ is } A_{i2} \quad (1)$$

$$\text{THEN } y = f_i$$

$$\omega_{i1} = A_{i1}(x_1^0) \quad (2)$$

$$\omega_{i2} = A_{i2}(x_2^0) \quad (3)$$

$$\mu_i = \omega_{i1} \cdot \omega_{i2}, \quad (i = 1, 2, \cdots, n) \quad (4)$$

$$y = \sum_{i=1}^{n} \mu_i \cdot f_i \quad (5)$$

人間にとって分かりやすいものであるが、1つ1つのルールの記述能力は小さい。そのため、この推論法を用いてファジィ制御器を構成するには、前件部メンバシップ関数の数を、ある程度増やすことが必要となる。しかし、同定可能なルール数は、2入力の制御器の場合、前件部メンバシップ関数の数の2乗に比例して増加するので、前件部メンバシップ関数の数を増やすことは、好ましくない。

そこで、本論文では1つ1つのルールの記述能力が優れる、後件部が線形式表現[6]されるファジィ推論法を用いる方法を提案する。これは2入力 (x_1^0, x_2^0) 1出力 (y^*) の制御器で、
制御器への入力が (x_1^0, x_2^0) の場合、次のように表される。

$$R^i : \text{IF } x_1 \text{ is } A_{i1} \text{ and } x_2 \text{ is } A_{i2} \quad \text{THEN } y_i = c_{i1} \cdot x_1 + c_{i2} \cdot x_2 + c_{i3} \tag{1}$$

$$\omega_{i1} = A_{i1}(x_1^0) \tag{2}$$

$$\omega_{i2} = A_{i2}(x_2^0) \tag{3}$$

$$\mu_i = \omega_{i1} \cdot \omega_{i2}, \quad (i = 1, 2, \cdots, n) \tag{4}$$

$$y^* = \frac{\sum_{i=1}^{n} \mu_i \cdot y_i}{\sum_{i=1}^{n} \mu_i} \tag{5}$$

ここで、R^i はi番目の制御ルール、ω_{i1}、ω_{i2} は R^i の前件部の入力 (x_1^0, x_2^0) のそれぞれに対する適合度、μ_i は R^i の前件部適合度、nは全ルール数である。また、A_{i1}、A_{i2} は「正で大」「負で小」といった言語的表現を表すファジィ変数 A^k を指し、よく用いられるものでは台形型[5]があるが、ここでは次式で表される疑似台形型を用いる。

$$A^k(x) = \frac{1}{1 + \exp(-wg_{k1}(x - wc_{k1}))} - \frac{1}{1 + \exp(-wg_{k2}(x - wc_{k2}))}, \quad (k = 1, \cdots, m) \tag{6}$$

ここで wg_{kj} はメンバシップ関数の傾きを決定し、wc_{kj} は中心位置を決定する。式(6)を見て分かるように、1つ1つのメンバシップ関数は2つのシグモイド関数の差によって構成される。ただし、両端のメンバシップ関数については、次式のように1つのシグモイド関数によって表現される。

$$A^k(x) = \frac{1}{1 + \exp(-wg_{k1}(x - wc_{k1}))} \tag{7}$$

Fig. 1 に m = 3 とした場合のメンバシップ関数の例を示す。

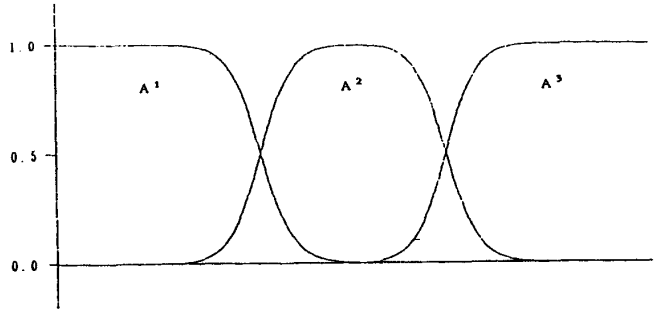


Fig. 1 Membership functions in premise

ここで、wc_{kj} の値によっては、$A^k(x) < 0$ となることがあるが、その場合、$A^k(x) = 0$ とする。

2．2　制御器の構成

2章1節で述べたファジィ推論法に基づいて、2入力1出力の場合の制御器は Fig. 2 の様に構成される。

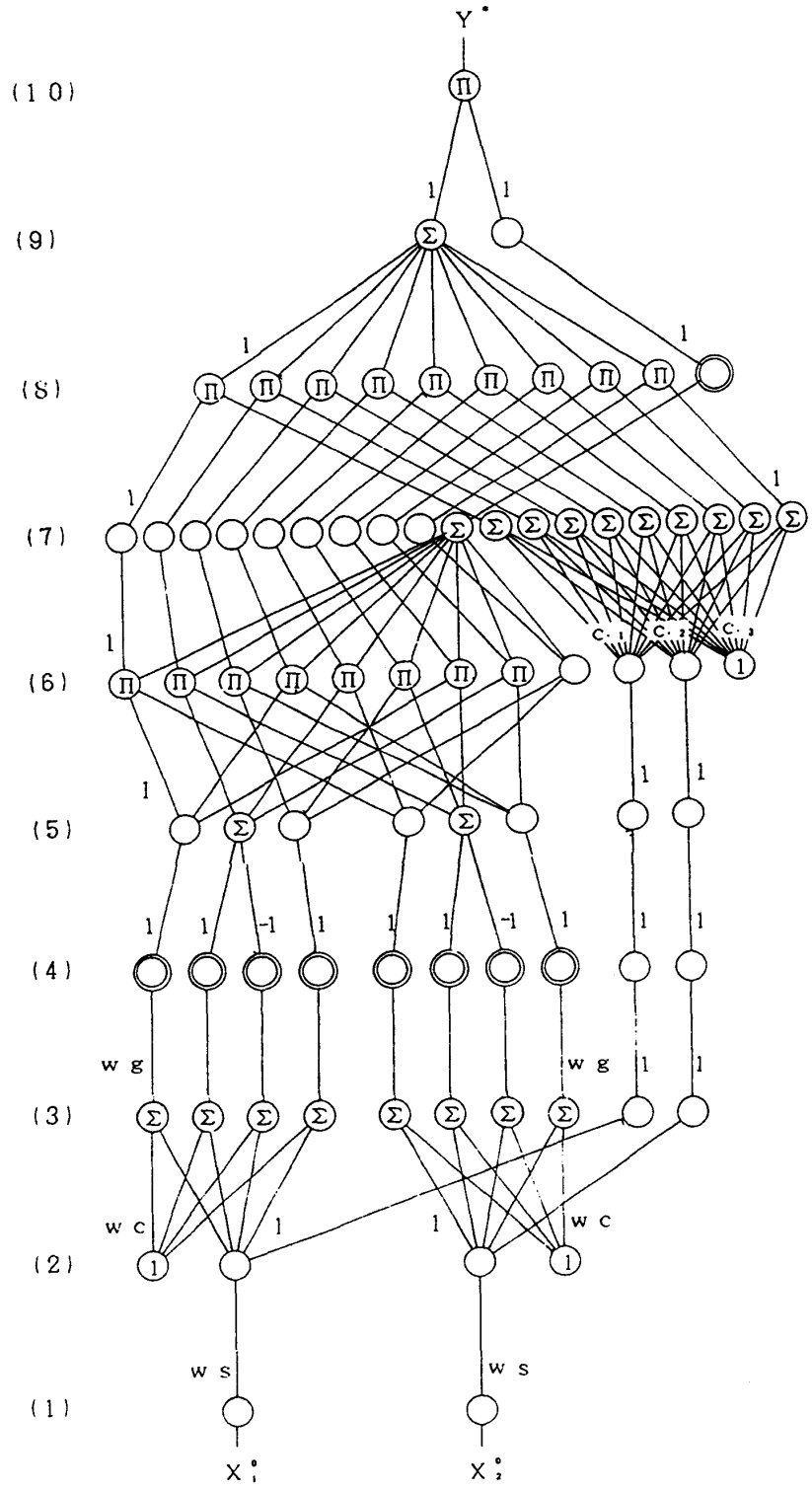


Fig. 2　Fuzzy controller with neural network

Fig. 2 で用いられる記号は以下の通りである。

○：内部関数がないユニット（入力がそのまま出力）

◎：内部関数を持つユニット

Σ：入力が線形和のユニット

Π：入力が積であるユニット

①：常に 1 を出力するユニット

1, −1：学習が行われない結合

ws, wc, wg, c ：学習が行われる結合

ここで、第 4 層、第 8 層における内部関数はそれぞれ、

$$f_4(x) = \frac{1}{1 + \exp(-x)} \tag{8}$$

$$f_8(x) = \frac{1}{x} \tag{9}$$

である。

また、Fig. 2 におけるファジィ推論の流れを見てみると、

（Ⅰ）第 1 層に加えられた入力 x_j^0 は、結合荷重 ws_j によって規格化される。

（Ⅱ）第 3 層では（Ⅰ）で規格化された入力に対して、バイアスである結合荷重 wc_{ij} が加えられる。

（Ⅲ）さらに wg_{ij} を掛けたものが第 4 層のシグモイド関数の入力となる。

（Ⅳ）第 4 層はシグモイド関数を内部関数として持ち、これによりメンバシップ関数が表現される。この出力が式(2)(3)の ω_{ij} である。

（Ⅴ）第 5 層の前件部部分では（Ⅳ）で求められた ω_{ij} をもとに、各ルールの μ_i が求められる。後件部部分では入力 x_j^0 に後件部の係数である c_{ij} が掛けられる。

（Ⅵ）第 6 層で各ルールの μ_i の和が求められる。

（Ⅶ）第 7 層では式(5)の分母部分が求められ、後件部の y_i が求められる。

（Ⅷ）第 8 層では式(5)の分子部分が求められ、分母部分では内部関数によりその逆数がとられる。

（Ⅸ）第 9 層では式(5)の分子の総和が求められる。

（Ⅹ）第10層で出力 y^* が求められる。

2．3　学習法

ネットワークは以下に示されるバックプロパゲーション法によって学習が行われる。まず、次式によって修正量 δ が求められる。

$$\text{出力層：} \delta_j^{(n)} = (t_j - O_j^{(n)}) f'(I_j^{(n)}) \tag{10}$$

$$\text{中間層：} \delta_j^{(n)} = f'(I_j^{(n)}) \sum_{k=1}^{u(n+1)} \delta_k^{(n+1)} w_{jk}^{(n)} \tag{11}$$

ここで、O は各ユニットの出力、I は各ユニットへの入力であり、次式にて求められる。

$$O_j^{(n)} = f(I_j^{(n)}) \tag{12}$$

$$\Sigma\text{結合：}\ I_k^{(n+1)} = \sum_{j=1}^{u(n)} w_{jk}^{(n)} O_j^{(n)} \tag{13}$$

$$\Pi\text{結合：}\ I_k^{(n+1)} = \prod_{j=1}^{u(n)} w_{jk}^{(n)} O_j^{(n)} \tag{14}$$

式(14)のΠ結合が行われるとき、式(11)の修正量は

$$\delta_j^{(n)} = f'(I_j^{(n)}) \sum_{k=1}^{u(n+1)} \delta_k^{(n+1)} w_{jk}^{(n)} \left(\prod_{i=1,\ i\neq j}^{u(n)} w_{ik}^{(n)} O_i^{(n)} \right) \tag{15}$$

となる。また、f(x)は内部関数である、内部関数を持つものについては式(8)、(9)で与えられ、内部関数を持たないものは入力がそのまま出力となる。また、f'(x)は内部関数の導関数であり、内部関数を持たないものは、1となる。すなわち、内部関数を持たないものは

$$f(x) = x \tag{16}$$

$$f'(x) = 1 \tag{17}$$

となる。次に式(10)(11)(15)によって求めた修正量δを用いて式(18)により結合荷重を更新する。

$$w_{jk}^{(n)} = w_{jk}^{(n)} + \eta\, \delta_k^{(n+1)} O_j^{(n)} \tag{18}$$

ただし、ws_jの学習については、入力変数によって、第1層のユニットの出力とその規格化定数であるws_jが大きく異なる可能性があるので、式(18)を用いると、その変化率が一様にならず学習率を個別に設定する必要が生じる。それ故、次式によって重みを更新する。

$$w_{jk}^{(n)} = w_{jk}^{(n)} (1 + \eta\, \delta_k^{(n+1)} w_{jk}^{(n)} O_j^{(n)}) \tag{19}$$

3．シミュレーション

ここでは、無駄時間＋1次遅れの系を制御対象としてシミュレーションを行う。Fig. 3は連続時間におけるシステム構成である。シミュレーションは、このシステムに対して、サンプリング間隔T＝0.1にて、離散時間による制御を行った。

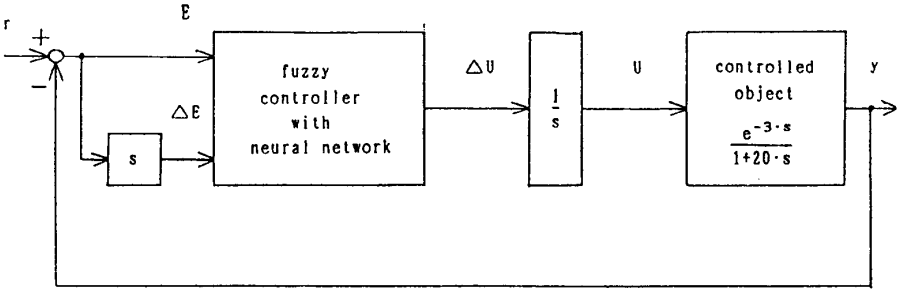


Fig. 3 Control system in continuos time

まず、制御器は2入力(E, △E) 1出力(△U)として構成した。ここで、Eは目標値rと制御対象の出力yとの誤差、△EはE変化分、△Uは操作量Uの変化分である。また、前件部メンバシップ関数は、E、△Eともに5分割とした。

本制御器は、文献4)に示される制御器（以下、制御器Ⅰと呼ぶ）を改善し、より優れた制御器を構成することが目的であるので、比較対象として、この制御器Ⅰを用いる。制御器Ⅰは、本論文で提案する制御器（以下、制御器Ⅱと呼ぶ）と同様、2入力(E, △E) 1出力(△U)で構成されるが、前件部メンバシップ関数は、E、△Eともに9分割されたものである。

つぎに、学習は、Fig. 3における制御対象に対して、良好な制御結果が得られるようチューニングされた従来型のファジィ制御器（学習用制御器）を対象に行った。そのとき用いたデータは、Fig. 4に示す225個のデータである。

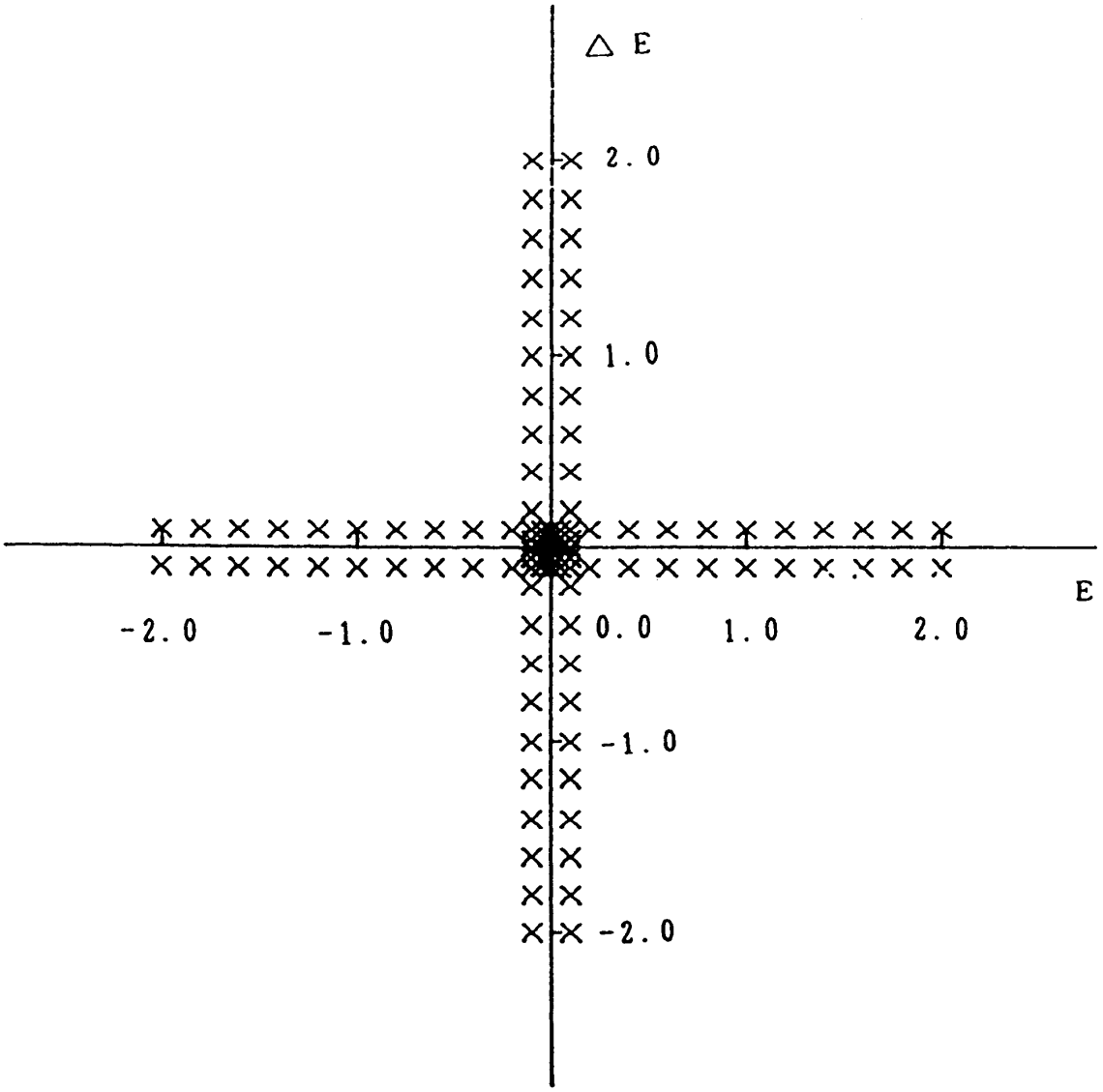
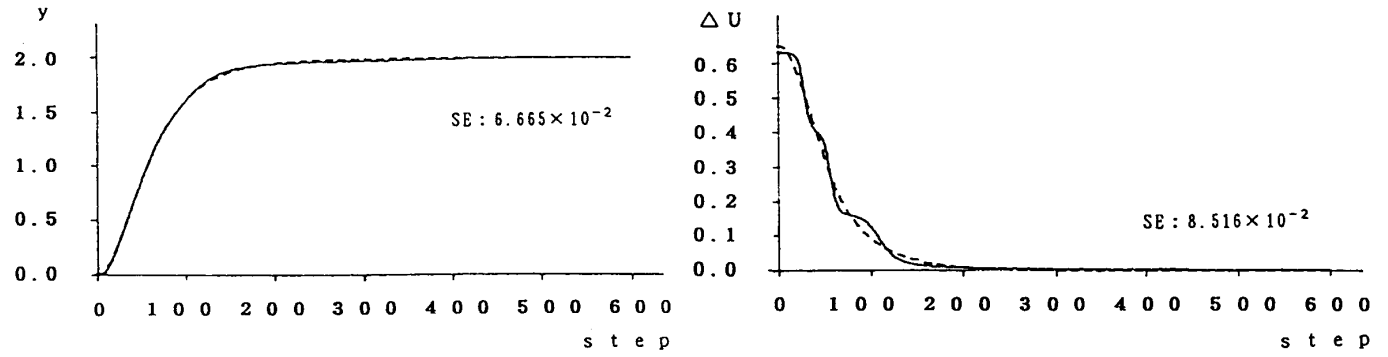
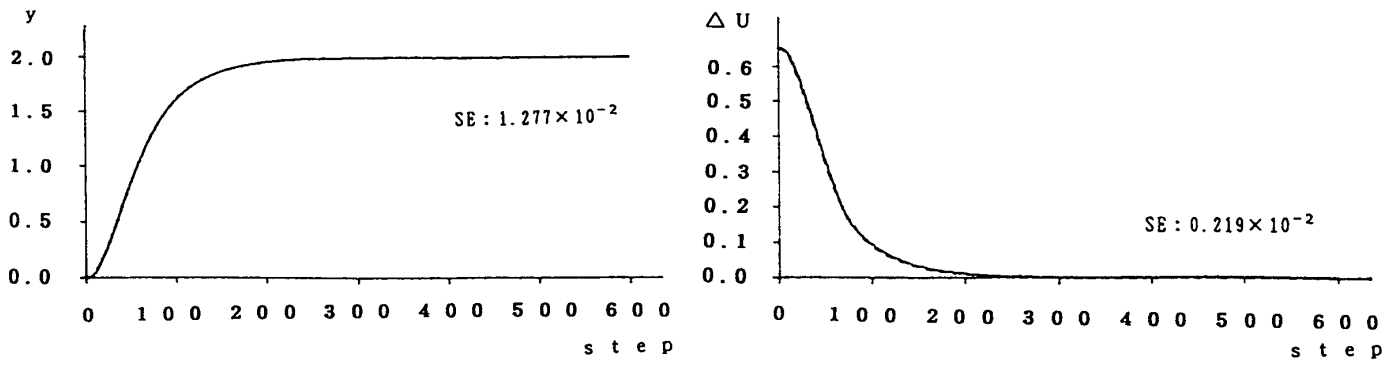


Fig. 4 Learning data

Fig. 4に示す225個のデータを1パターンとして、制御器Ⅰについては、5000回の学習を行った後に、制御器Ⅱについては1000回の学習を行った後に、シミュレーションを行った結果をFig. 5に示す。点線は、教師信号用の従来型ファジィ制御器である。



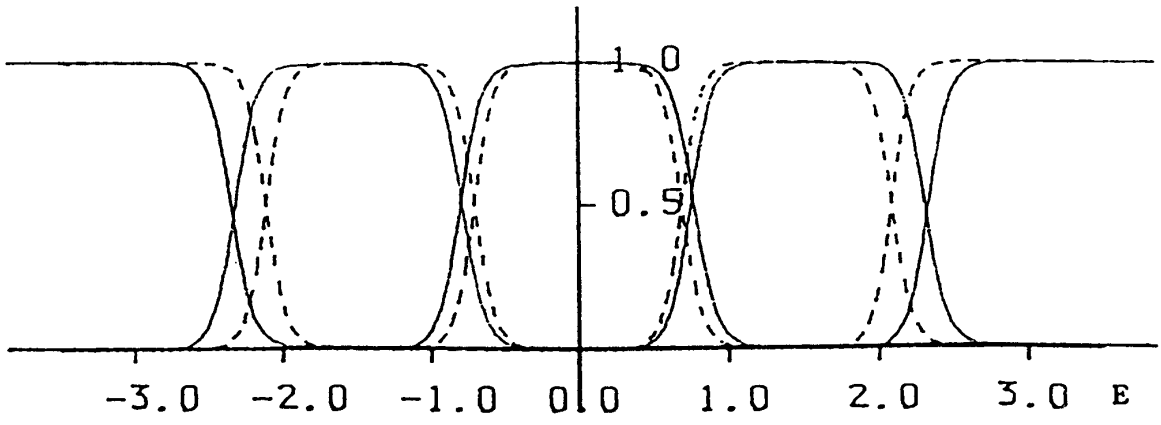
(a-1) Response of system (Cntroller Ⅰ) (b-1) Output of controller (Cntroller Ⅰ)



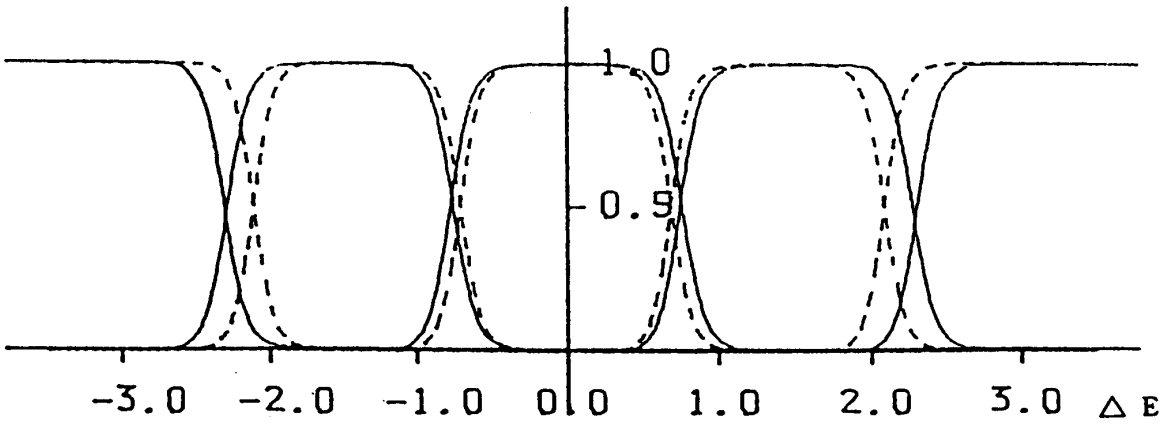
(a-2) Response of system (Cntroller Ⅱ) (b-2) Output of controller (Cntroller Ⅱ)

Fig. 5 Simulation results (SE: Square Eroor)

Fig. 5 (a)は、制御対象の出力であり、(a-1)が制御器Ⅰ、(a-2)が制御器Ⅱのものであり、(b)は制御器の出力すなわち、操作量の変化分△U の波形である。ここで、Fig. 5 における点線の波形はいずれも学習用制制御器の波形である。また、図中2乗誤差とあるのは、0～600step における、学習制御器と、制御器Ⅰ、Ⅱの誤差の2乗和である。Fig. 5 (a)を見ると、制御器Ⅰ、Ⅱ共に良好な制御結果を得ているが、2乗誤差を見ると制御器Ⅱは制御器Ⅰのおよそ1/5と改善されていることが分かる。つぎに、Fig. 5 (b)を見ると、制御器Ⅰはかなり不安定な出力波形をしているが、制御器Ⅱでは良好な結果が得られていることが分かる。2乗誤差についても、制御器Ⅱは制御器Ⅰのおよそ1/40と大幅に改善されていることが分かる。Fig. 6 は、学習前後の前件部メンバシップ関数である。



(a) E



(b) △E

Fig. 6 Membership functions in premise

また、Table 1は得られた制御規則である。0.0とあるのは学習が行われていないことを意味する。これは、Fig. 4 を見て分かるように、学習データが存在しないところがあるからである。

Table 1 Control rules

PL :Positive Large NL :Negative Large

PS :Positive Small NS :Negative Small

ZR :ZeRo

$\Delta U = c_1 x\ E + c_2 x \Delta E + c_3$ $(x10^{-1})$		△E				
		PL	PS	ZR	NS	NL
E	PL	0.0 0.0 0.0	0.0000 0.0000 0.0000	0.5330 0.0000 0.2752	0.0000 0.0000 0.0000	0.0 0.0 0.0
	PS	0.0000 0.0000 0.0000	0.0002 0.0011 0.0008	4.0732 0.0002 -0.8892	0.0003 -0.0012 0.0012	0.0 0.0 0.0
	ZR	-0.0032 0.7098 0.3648	1.5106 4.0666 -0.9984	2.6452 2.6819 0.0001	1.5954 4.0556 0.9818	0.0145 0.2872 -0.1500
	NS	0.0000 0.0000 0.0000	0.0002 -0.0008 -0.0009	3.9246 0.0001 0.6427	0.0002 0.0013 -0.0007	0.0000 0.0000 -0.0000
	NL	0.0 0.0 0.0	0.0000 0.0000 0.0000	0.5272 0.0000 -0.2722	0.0000 0.0000 -0.0000	0.0 0.0 0.0

4．まとめ

以上、本論文で提案するニューラルネットワークを用いたファジィ制御器について、そのファジィ推論法、構成法、学習法について述べ、無駄時間＋1次遅れの系を制御対象とした、シミュレーションよってその有効性を示した。その結果、簡略化されたファジィ推論法を用いた文献4)の制御器よりも、本論文で、提案した制御器の方が、より少ない制御ルールで、より確実にエキスパートの持つ制御規則を抽出できたと考えられる。

今後の課題としては、まず、Table 1 を見て分かるように、学習データによってまったく不要なルールの組み合わせが存在する事があるので、不要なルールを学習の段階で、自動的に削除することがあげらる。また、今回のシミュレーションは、学習データをエキスパートの操作パターンとせず、従来型のファジィ制御器の操作パターンを用いているので、実システムへの応用が考えられる。

参 考 文 献

1）林勲，ほか：ニューラルネット駆動型ファジィ推論による倒立振子の学習制御，第5回ファジィシステムシンポジウム講演論文集，183/188(1989)
2）林陽一，ほか：ニューラルネットワークを用いたファジィＩＦ～ＴＨＥＮルールの自動抽出，電学論Ｃ，108，198/206(1990)
3）ニューラルネットワークを用いた自己組織化的ファジィ制御器の構成，計測自動制御学会論文集、Vol. 26，No.8，862/869(1990)
4）堀川慎一，ほか：ニューラルネットワークによる学習型ファジィ制御器，計測自動制御学会論文集，Vol. 23, No. 6，208/215 (1991)
5）水本雅晴：ファジィ制御向きのファジィ推論法，計測と制御，Vol. 28，No. 11 959/963(1989)
6）菅野道夫：ファジィ制御，日刊工業新聞社(1988)