An Efficient Algorithm for Automatically Generating Multivariable Fuzzy Systems by Fourier Series Method

Liang Chen and Naoyuki Tokuda, *Member, IEEE*

***Abstract*—By exploiting the Fourier series expansion, we have developed a new constructive method of automatically generating a multivariable fuzzy inference system from any given sample sets with the resulting multivariable function being constructed within any specified precision to the original sample set. The given sample sets are first decomposed into a cluster of simpler sample sets such that a single input fuzzy system is constructed readily for a sample set extracted directly from the cluster independent of the other variables. Once the relevant fuzzy rules and membership functions are constructed for each of the variables completely independent of the other variables, the resulting decomposed fuzzy rules and membership functions are integrated back into the fuzzy system appropriate for the original sample set requiring only a moderate cost of computation in the required decomposition and composition processes.**

After proving two basic theorems which we need to ensure the validity of the decomposition and composition processes of the system construction, we have demonstrated a constructive algorithm of a multivariable input, namely, a multivariable output fuzzy system. Exploiting an implicit error bound analysis available at each of the construction steps, the present Fourier method is capable of implementing a more stable fuzzy system than the power series expansion method of ParNeuFuz [3] and PolyNeuFuz [4], covering and implementing a wider range of more robust applications.

***Index Terms*—Fourier series, fuzzy system, neural network.**

I. Introduction

IT is well known [1] that fuzzy systems are extremely effective in approximating any continuous function on a compact set. Taking advantage of its additional unique capability of capturing an approximate, qualitative aspect of human knowledge and reasoning needed in many of expertise-dominated systems [14], the fuzzy system is found to be extremely effective in controlling some complex control systems, providing a powerful alternative to the conventional control system particularly for complex and/or ill-defined systems including an increasingly wider range of applications in industrial applications and household appliances, as well as in financial analysis [9], [11], [12], [16], [20], [22], [24].

Manuscript received August 27, 2000; revised August 30, 2001 and March 5, 2002. This paper was recommended by Associate Editor A. Kandel.

L. Chen is currently with the Department of Math and Computer Science, University of Northern British Columbia, BC V2N 4Z9, Canada (e-mail: chenl@unbc.ca).

N. Tokuda is with the Research and Development Center, Sunflare Company, Shinjuku-ku, Tokyo 160-0004, Japan, and also with the Department of Computer Science, Utsunomiya University, Utsunomiya 321-8505, Japan (e-mail: tokuda_n@sunflare.co.jp).

Publisher Item Identifier S 1083-4419(02)05272-X.

Because the performance of a fuzzy system depends critically on the depth of experts' understanding of the domain knowledge, as in many expert systems, a serious bottleneck problem arises due to the soaring cost of having cooperation from first-class experts in the field for system development. This motivates an intensive study on automatic generation of fuzzy systems and the subject has often been studied as a data-driven, self-adaptive problem. We refer to [3] for a review of the related works on this topic.

One of the biggest problems we immediately encounter in solving the self-adaptive problem relates to what we call a *dimensional explosion predicament* [4], [12], which is associated with an exponential explosion in the required computational time as the number of variables increases. This is because the relationship among the variables of the fuzzy rules and membership functions is often quite complicated and is, more often than not, nonlinear. Because of this, almost all of the existing works available can only deal with a relatively small number of variables. As far as we know, none of these algorithms can deal with more than five input variables. For example, the well-known fuzzy logic design kit NeuFuz4, by Kahn *et al.* [17]–[19], can only solve the problem of at most, four variables. Even for very limited input variables, most of them can only deal with "straight" membership functions, such as trapezoidal-shaped membership functions, so that the resulting controllers represent a linear control. Furthermore, very few [15], [17]–[19] can deal with continuous membership functions as represented by Gaussian membership functions so that nonlinear effects of controllers can be taken into account [13].

Chen *et al.* [4]–[8] presented a new algorithm, PolyNeuFuz, based on polynomial approximation. PolyNeuFuz was later improved to ParNeuFuz [3] which, in principle, can be run in parallel processing if one takes advantage of the independence of the input variables. Like the present Fourier-series-based approach, both PolyNeuFuz and ParNeuFuz solve the problem of generating multivariable fuzzy systems by decomposing the problem to a solution of single input, multiple outputs fuzzy systems. In place of Fourier series decomposition, PolyNeuFuz and ParNeuFuz exploit a polynomial expansion in approximating the given sample sets. Because of the decomposition and composition processes introduced, the time complexity of the algorithms depends only linearly on the number of variables used, so that the algorithms can be applied to generate a fuzzy system with a large number of variables without much problem, thus resolving the dimensional explosion predicament. However, an error estimation or prediction at several key steps of the computation scheme is not easy for the PolyNeuFuz and ParNeuFuz methods.

The purpose of this paper is to develop a new Fourier-series-based automatic generation scheme for a multivariable fuzzy inference system capable of providing accurate error estimates at each step of the computational scheme consistent to the overall error bounds needed, while at the same time avoiding the dimensional explosion predicament. Following PolyNeuFuz and ParNeuFuz [4], [3], the fuzzy rules and membership functions for each variable will be developed completely independent of the other variables. To do so, we first decompose the sample set, e.g., Λ, into a cluster of sample sets associated with each of the given input variables, and individually compute the fuzzy rules and membership functions for each variable by solving a single input, multiple outputs fuzzy system extracted from the set cluster. The resulting fuzzy rules and membership functions are composed and integrated back into the fuzzy system appropriate for the original sample set Λ with a minimal computational cost. Taking an advantage of the fact that the decomposition is based on a Fourier series, a careful error analysis shows how an overall system error can be related to errors at each stage of decomposition and composition so that we are now able to specify the final precision of the resulting fuzzy system on the original sample set consistent to the overall system errors specified. Unlike its predecessor PolyNeuFuz [3], which has depended on a trial and error basis with possible "over accuracy" computations, our FouNeuFuz method considerably reduces the overall computational time by avoiding the process of trial and error processes. An additional advantage of the present approach relates to a further reduction in computational time by implementing the present scheme on a parallel processor, because the complete independence of the input variables is ensured for computing the fuzzy rules and membership functions for each variable, as in ParNeuFuz [3].

In Section II, we prove two theorems which form the theoretical foundation of the present method, ensuring the validity of the present Fourier-series-based decomposition and composition process. In Section III, a constructive algorithm is given for computing the fuzzy rules and membership functions from the given sample set. In Section IV, we demonstrate how our method is capable of simulating 8-variable functions with considerable ease and accuracy than the conventional methods. The conclusion is given in Section V.

II. Theoretical Foundation

The two theorems presented in this section form the foundation of the present method. Theorem 1 ensures the decomposition of the original sample set Λ into a number of simpler sample sets, which form the basis for the decomposition process of our method. Theorem 2 ensures the construction and composition procedure of decomposed single input fuzzy systems to recover the fuzzy system on the original sample set Λ. Theorem 2 is proved in terms of two lemmas, namely, Lemma 1 and Lemma 2. Lemma 1 ensures the construction process of a fuzzy system characterized by a multiple inputs, single output sample set from the decomposed single input single output sample set of Theorem 1, while Lemma 2 ensures the reconstruction process of an overall fuzzy system satisfying the original specified overall sample set.

Throughout the paper, we denote the number of samples by N, and the number of input variables by m, with both N and m being finite. Fourier series theory [10] shows that any continuous function of multivariables can be uniformly approximated by the sum of the Fourier series.

Theorem 1: For any sample set $\Lambda = \{(\bar{X}_j, Y_j) \mid j = 1,2,\ldots,N\}$, where $\bar{X}_j = (X_{1,j}, X_{2,j}, \ldots, X_{m,j})$ and $L_i \le X_i \le U_i$ for all i (L_i and U_i denote, respectively, the minimum and maximum of real variable X_i, with $L_i \ne U_i$ being assumed), there exists a set $a = \{a_{\bar{s}} \ne 0 \mid \bar{s} = (s_1, s_2, \ldots, s_{2m}) \in \mathcal{S}, s_2, s_4, \ldots, s_{2m} \in \{0,1\}$, with $s_1, s_3, \ldots, s_{2m-1}$ being nonnegative integers$\}$, and a sample set cluster $\mathcal{A} = \{\mathcal{A}_{\bar{s}} = \{(\bar{X}_j, f_{\bar{s},1,j}, f_{\bar{s},2,j}, \ldots, f_{\bar{s},m,j}) \mid j = 1,2,\ldots,N\} \mid \bar{s} = (s_1, s_2, \ldots, s_{2m}) \in \mathcal{S}\}$, such that

$$Y_j = \sum_{\bar{s}\in\mathcal{S}} (a_{\bar{s}} \cdot f_{\bar{s},1,j} f_{\bar{s},2,j} \ldots f_{\bar{s},m,j}) \tag{1}$$

where

$$f_{\bar{s},i,j} = \cos\left(\frac{2\pi s_{2i-1}}{U_i - L_i} \cdot \left(X_{i,j} - \frac{U_i - L_i}{2}\right) + s_{2i} \cdot \frac{\pi}{2}\right). \tag{2}$$

Proof: Fourier series theory [10] ensures the existence of Fourier series $F_{\bar{s}}(\bar{X}) = a_{\bar{s}} \prod_{i=1}^{m} \cos((2\pi s_{2i-1})/(U_i - L_i) \cdot (X_i - (U_i - L_i)/2) + s_{2i} \cdot (\pi/2))$, $\bar{s} \in \mathcal{S}$, such that $\sum_{\bar{s}\in\mathcal{S}} F_{\bar{s}}(\bar{X}_j) = Y_j$ ($j = 1,2,\ldots,N$). The equality is ensured for discrete input variables X and output Y with N and m being finite.

Write $f_{\bar{s},i,j} = \cos((2\pi s_{2i-1})/(U_i - L_i) \cdot (X_{i,j} - (U_i - L_i)/2) + s_{2i} \cdot (\pi/2))$. If we let the $a_{\bar{s}}$s be the coefficients of the orthogonal functions $\prod_{i=1}^{m} \cos((2\pi s_{2i-1})/(U_i - L_i) \cdot (X_i - (U_i - L_i)/2) + s_{2i} \cdot (\pi/2))$ (with $|a_{\bar{s}}|$ bounded by some finite constant say, Q), we have

$$F_{\bar{s}}(\bar{X}) = a_{\bar{s}} \prod_{i=1}^{m} f_{\bar{s},i,j}$$

and

$$Y_j = \sum_{\bar{s}\in\mathcal{S}} F_{\bar{s}}(\bar{X}_j).$$

■

Theorem 2: Suppose that the single input, multiple output sample set $\tilde{B}_i = \{(X_{i,j}, \{f_{\bar{s},i,j} | \bar{s} \in \mathcal{S}\}) | j = 1,2,\ldots,N\}$ ($|f_{\bar{s},i,j}| \le 1$) can be described by the fuzzy system MFS_i of Table I. Then the fuzzy sample set $\mathcal{D} = \{(\bar{X}_j, \sum_{\bar{s}\in\mathcal{S}} (a_{\bar{s}} \cdot f_{\bar{s},1,j} f_{\bar{s},2,j} \ldots f_{\bar{s},m,j})) \mid j = 1,2,\ldots,N\}$ can be described by the fuzzy system MFS in Table I.

Proof: Because the fuzzy system MFS_i of Theorem 2 can be regarded as $|\mathcal{S}|$ single input single output fuzzy systems $\mathrm{FS}_{\bar{s},i}$ (for $\bar{s} \in \mathcal{S}$), each describing a single input single output sample set $\mathcal{B}_{\bar{s},i} = \{(X_{i,j}, f_{\bar{s},i,j}) \mid j = 1,2,\ldots,N\}$, Theorem 2 can be proved in terms of the following Lemma 1 and Lemma 2, the proofs of which are given in Appendix. ■

Lemma 1: Suppose that the sample set $\mathcal{B}_{\bar{s},i} = \{(X_{i,j}, f_{\bar{s},i,j}) \,|\, j = 1, 2, \ldots, N\}$ ($|f_{\bar{s},i,j}| \leq 1$) can be described by the fuzzy system $\mathrm{FS}_{\bar{s},i}$ ($i = 1, 2, \ldots, m$) of Table II. Then, the sample set $\mathcal{B}_{\bar{s}} = \{(\bar{X}_j, f_{\bar{s},1,j} f_{\bar{s},2,j} \cdots f_{\bar{s},m,j}) \,|\, j = 1, 2, \ldots, N\}$ can be described by the fuzzy system $\mathrm{FS}_{\bar{s},i}$ of Table II.

Lemma 2: Suppose that for each $\bar{s} \in \mathcal{S}$, the fuzzy system $FS_{\bar{s}}$ of Table III describes the sample set $\mathcal{C}_{\bar{s}} = \{(\bar{X}_j, g_{\bar{s}j}) \,|\, j = 1, 2, \ldots, N\}$ ($g_{\bar{s}j} \in [-1, 1]$). Then, the fuzzy system FSS of Table III can now describe the sample set $\mathcal{D} = \{(\bar{X}_j, \sum_{\bar{s} \in \mathcal{S}} (a_{\bar{s}} \cdot g_{\bar{s}j})) \,|\, j = 1, 2, \ldots, N\}$.

Lemma 1 shows how we can construct, for each given $\bar{s} \in \mathcal{S}$, a fuzzy system $\mathrm{FS}_{\bar{s}}$ for the sample set $\mathcal{B}_{\bar{s}} = \{(\bar{X}_j, f_{\bar{s},1,j} f_{\bar{s},2,j} \cdots f_{\bar{s},m,j}) \,|\, j = 1, 2, \ldots, N\}$ from the fuzzy systems $\mathrm{FS}_{\bar{s},i}$ (for $i = 1, 2, \ldots, m$), describing $\mathcal{B}_{\bar{s},i}$ (for $i = 1, 2, \ldots, m$). Writing $g_{\bar{s}j} = f_{\bar{s},1,j} f_{\bar{s},2,j} \cdots f_{\bar{s},m,j}$, we note readily that $\mathcal{B}_{\bar{s}} = \{(\bar{X}_j, g_{\bar{s}j}) \,|\, j = 1, 2, \ldots, N\}$ of Lemma 1 is identical to $\mathcal{C}_{\bar{s}}$ of Lemma 2. Now, Lemma 2 completes the proof of Theorem 2 by showing how we obtain the final fuzzy system describing the given sample set $\mathcal{D}$; this can be done by combining and integrating all the fuzzy systems for sample sets $\mathcal{B}_{\bar{s}}$ for $\bar{s} \in \mathcal{S}$ of Lemma 1 with due consideration paid to error bound specifcations. Assigning the maximum error bound and the sum of errors bound of $\mathrm{FS}_{\bar{s}}$ of Table II to those of $\mathrm{FS}_{\bar{s}}$ of Table III, i.e., $\varepsilon_1' = m\varepsilon_1' + O((1/2)(m\varepsilon_1)^2)$ and $\varepsilon_2' = (1 + (1/2)m\varepsilon_1 + O((1/6)(m\varepsilon_1)^2))m\varepsilon_2$, the maximum error bound and the sum of errors bound of FSS in Table III will match those of MFS, thus proving Theorem 2.

Note that $\mathcal{D}$ of Lemma 2 is equivalent to Λ of Theorem 1, as long as (1) holds. ■

III. Constructive Algorithm

Theorem 1 and Theorem 2 ensure the main ideas of our scheme for an automatic generation of fuzzy systems. Actually, Theorem 1 ensures the existence of a set cluster $\mathcal{A}$ decomposed from original sample set Λ. Theorem 2 shows that, once the fuzzy systems each with only one variable X_i are computed on a sample set $\tilde{B}_i$ extracted directly from the set cluster $\mathcal{A}$, we can construct the fuzzy system appropriate for Λ by following the steps of Theorem 2.

In this section, we will first establish a decomposition method for obtaining the set cluster $\mathcal{A}$, and then establish a method for obtaining a fuzzy system with single variable on the sample set $\tilde{B}_i$. We will then show how the whole constructive algorithm for computing a fuzzy system on the original sample set Λ can be completed.

A. Decomposition Method

Knowledge of the sample set as well as on the smoothness of an object surface plays an important role in our analysis. Suppose that the object system does not include higher components than $2\pi\tau_i/(U_i - L_i)$ for each variable X_i. Then we can select an $\mathcal{S} = \{(s_1, s_2, \ldots, s_{2m})\}$ where $s_1 \leq \tau_1, s_3 \leq \tau_2, \ldots s_{2m-1} \leq$

TABLE I
Fuzzy Systems MFS_i and MFS

	MFS_i $(i = 1, 2, \cdots, m)$	
Membership Functions	For each $t_i, t_i = 1, 2, \cdots, \tau_i$($\tau_i$ is the number of fuzzy intervels of variable x_i), the membership function of X_i at fuzzy interval S_{i,t_i} is μ_{i,t_i}	For each pai the members
Fuzzy Rules	For each t_i, $t_i = 1, 2, \cdots, \tau_i$, if X_i is S_{i,t_i} then the $\bar{s}$-th output$=O_{\bar{s},i,t_i}$	For each m- if X_1 is S_{1,t_1} *Output* $= O_t$
Defuzzification Method	$\bar{s}$-*th Output* $O_{fuzzy} = \sum_{t_i=1}^{\tau_i} \mu_{i,t_i}(X_i) \cdot O_{\bar{s},i,t_i}$	$O_{fuzzy} =$
Maximum Error	$\max_{j \in \{1,2,\cdots,N\}} \lvert O_{fuzzy}(X_{i,j}) - f_{\bar{s},i,j}) \rvert \leq \varepsilon_1$ $(\varepsilon_1 < 1/m)$	$\max_{j \in \{1,2,\cdots,N\}}$ $\leq Q\lvert\mathcal{S}\rvert(m\varepsilon_1$
Sum of Errors	$\sum_{j=1}^{N} \lvert O_{fuzzy}(X_{i,j}) - f_{\bar{s},i,j}) \rvert \leq \varepsilon_2$ $(\varepsilon_1 < \varepsilon_2 < N\varepsilon_1)$	$\sum_{j=1}^{N} \lvert O_{fuzzy}($ $\leq (1 + \frac{1}{2}m\varepsilon_1$

TABLE II
Fuzzy Systems $\mathrm{FS}_{\bar{s},i}$ and $\mathrm{FS}_{\bar{s}}$

	$\mathrm{FS}_{\bar{s},i}, (i = 1, 2, \cdots, m)$	
Membership Functions	For each $t_i, t_i = 1, 2, \cdots, \tau_i$, the membership function of X_i at fuzzy interval S_{i,t_i} is μ_{i,t_i}	For each pair of the membership
Fuzzy Rules	For each t_i, $t_i = 1, 2, \cdots, \tau_i$, if X_i is S_{i,t_i} then output$=O_{\bar{s},i,t_i}$	For each m-tupl if X_1 is S_{1,t_1} *an* *output* $O^{(\bar{s})}_{t_1,t_2,\cdots}$
Defuzzification Method	$O_{fuzzy} = \sum_{t_i=1}^{\tau_i} \mu_{i,t_i}(X_i) \cdot O_{\bar{s},i,t_i}$	$O_{fuzzy} = \sum_{t_i=1,2,\cdots;\ i=1,2,\cdots}$
Maximum Error	$\max_{j \in \{1,2,\cdots,N\}} \lvert O_{fuzzy}(X_{i,j}) - f_{\bar{s},i,j}) \rvert \leq \varepsilon_1$ $(\varepsilon_1 < 1/m)$	$\max_{j \in \{1,2,\cdots,N\}} \lvert O_{fu}$
Sum of Errors	$\sum_{j=1}^{N} \lvert O_{fuzzy}(X_{i,j}) - f_{\bar{s},i,j}) \rvert \leq \varepsilon_2$ $(\varepsilon_1 < \varepsilon_2 < N\varepsilon_1)$	$\sum_{j=1}^{N} \lvert O_{fuzzy}(\overline{X}_j)$

TABLE III
Fuzzy Systems $\mathrm{FS}_{\bar{s}}$ and FSS

	$\mathrm{FS}_{\bar{s}}$	
Membership Functions	For each pair of $(i, t_i), t_i = 1, 2, \cdots, \tau_i$, the membership function of X_i at fuzzy interval S_{i,t_i} is μ_{i,t_i}	For each pa the membe
Fuzzy Rules	For each m-tuple $(t_1, t_2, \cdots, t_m)$, $t_i = 1, 2, \cdots, \tau_i$, $i = 1, 2, \cdots, m$, if X_1 is S_{1,t_1} *and* X_2 is S_{2,t_2} *and* $\cdots$ *and* X_m is S_{m,t_m} then *output* $= O^{(\bar{s})}_{t_1,t_2,\cdots,t_m}$	For each m if X_1 is S_1, *output* $= O$
Defuzzification Method	$O_{fuzzy} = \sum_{\substack{t_i=1,2,\cdots,\tau_i \\ i=1,2,\cdots,m}} \left(\prod_{l=1}^{m} \mu_{l,t_l}(X_l)\right) \cdot O^{(\bar{s})}_{t_1,t_2,\cdots,t_m}$	$O_{fuzzy} =$
Maximum Error	$\max_{j \in \{1,2,\cdots,N\}} \lvert O_{fuzzy}(\overline{X}_j) - g_{\bar{s}j} \rvert \leq \varepsilon_1'$	$\max_{j \in \{1,2,\cdots,N\}}$
Sum of Errors	$\sum_{j=1}^{N} \lvert O_{fuzzy}(\overline{X}_j) - g_{\bar{s}j} \rvert \leq \varepsilon_2'$	$\sum_{j=1}^{N} \lvert O_{fuzzy}$

τ_m. An optimal approximation to the sample set can be obtained by minimizing the following function:

$$E = \frac{1}{2} \sum_{j=1}^{N} \left(Y_j - \sum_{\bar{s}=(s_1, s_2, \ldots, s_{2m}) \in \mathcal{S}} \left(a_{\bar{s}} \prod_{i=1}^{m} \cos\left(\frac{2\pi s_{2i-1}}{U_i - L_i} \right.\right.\right.$$

$$\cdot\left(X_{i,j}-\frac{U_i-L_i}{2}\right)+s_{2i}\cdot\frac{\pi}{2}\Bigg)\Bigg)\Bigg)^2. \quad (3)$$

This is equivalent to finding the roots of the equations $\partial E/\partial a_{\bar{s}} = 0$. Thus, $a_{\bar{s}}$ can be determined by determinants, as in [4]. We need some care in the process of collecting all terms of equal wave numbers into one term lest any two terms left over differ only in coefficients. For example, the two terms $a_{(0,0,1,0)}\cos(0)\cdot\cos((2\pi s_1)/(U_1-L1)\cdot(X_2-(U_1-L_1)/2))$ and $a_{(0,1,1,0)}\cos((\pi/2))\cdot\cos((2\pi s_1)/(U_1-L1)\cdot(X_2-(U_1-L_1)/2))$ should be combined into $a'_{(0,0,1,0)}\cos(0)\cdot\cos((2\pi s_1)/(U_1-L1)\cdot(X_2-(U_1-L_1)/2))$ before we determine the coefficients of the terms of Fourier series. Once this is done, if $\mathcal{S}=(\bar{s}_1,\bar{s}_2,\ldots,\bar{s}_\tau)$, with $\bar{s}_k=(s_{k_1},s_{k_2},\ldots,s_{k_{2m}})$, $k=1,2,\ldots,\tau$, the $a_{\bar{s}_k}$ can be expressed in the following simplified form by determinants:

$$a_{\bar{s}_k}=\frac{\left|\mathcal{T}^k_{p,q}\right|_{\tau\times\tau}}{\left|\mathcal{J}_{p,q}\right|_{\tau\times\tau}} \quad (4)$$

where (see the equation at the bottom of the next page).

In case the sample data fall exactly on the multidimensional grids, a standard fast Fourier transform (FFT) method [23] can be employed to speed up this process.

B. Generating Single Input, Multiple Outputs Fuzzy Systems

Consider the neural network in Fig. 1. Each of the link weights $b_{it,k}$ between the second-layer neuron and the third-layer neuron can be interpreted as a rule: "*if* X_i *is* $S_{i,t}$ *then the output of* f_k *is* $b_{it,k}$," then the relationship between the input and output of the neural network $f_k=b_{i1,k}\mu_{i1}(X_i)+b_{i2,k}\mu_{i2}(X_i)+\cdots+b_{i\tau_i,k}\mu_{i\tau_i}(X_i)$ can be described by the following fuzzy system S:

- Membership functions:
  For each t, where $t=1,2,\ldots,\tau_i$, the membership of fuzzy interval $S_{i,t}$ is $\mu_{i,t}$.
- Fuzzy rules:
  For each pair of (t,k), where $t=1,2,\ldots,\tau_i; k=1,2,\ldots,\tau$, if X_i is $S_{i,t}$ then the output of $f_k=b_{it,k}$.
- Defuzzification method:
  output of $f_k=\sum_{t=1}^{\tau_i}\mu_{i,t}(X_i)b_{it,k}$.

Thus, the task of obtaining the membership functions and fuzzy rules of each variable X_i reduces to training the neural network in Fig. 1 using the set $\tilde{B}_i$ as its sample set. The membership functions of each variable can either be of "straight" type or continuous type. A steepest descent (SD) method [2], [21] can be used to train these individual networks.

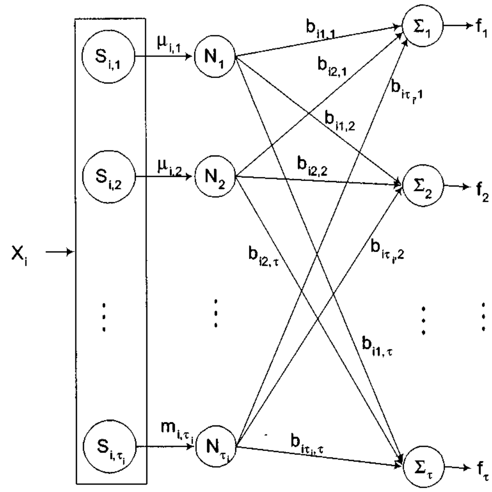

Fig. 1. Neural network generating fuzzy system of single input and multiple outputs.

C. Algorithm FouNeuFuz for Generating Multivariable Fuzzy System

Suppose that we need to develop the fuzzy system on sample set Λ with the maximum error bound and the sum of errors bound of ε_1 and ε_2, respectively, such that $\max_{j\in\{1,2,\ldots,N\}}|O_{\text{fuzzy}}(\bar{X}_j)-Y_j|\le\varepsilon_1$ and $\sum_{j=1}^N|O_{\text{fuzzy}}(\bar{X}_j)-Y_j|\le\varepsilon_2$.

1) Decompose the sample set Λ into the sample set cluster $\mathcal{A}=\{\mathcal{A}_{\bar{s}}=\{(\bar{X}_j,f_{\bar{s},1,j},f_{\bar{s},2,j},\ldots,f_{\bar{s},m,j})\,|\,j=1,2,\ldots,N\}\,|\,\bar{s}=(s_1,s_2,\ldots,s_{2m})\in\mathcal{S}\}$ by minimizing (3), and letting $f_{\bar{s},i,j}=\cos((2\pi s_{2i-1})/(U_i-L_i)\cdot(X_{i,j}-(U_i-L_i)/2)+s_{2i}\cdot(\pi/2))$; Calculate the maximum error and the total errors: $\varepsilon_{f1}=\max_{j\in\{1,2,\ldots,N\}}|Y_j-\sum_{\bar{s}\in\mathcal{S}}(a_{\bar{s}}\cdot f_{\bar{s},1,j}f_{\bar{s},2,j}\ldots f_{\bar{s},m,j}))|$, $\varepsilon_{f2}=\sum_{j=1}^N|Y_j-\sum_{\bar{s}\in\mathcal{S}}(a_{\bar{s}}\cdot f_{\bar{s},1,j}f_{\bar{s},2,j}\ldots f_{\bar{s},m,j}))|$. Make sure that $\varepsilon_{f1}\le\varepsilon_1$, and $\varepsilon_{f2}\le\varepsilon_2$. If not, add terms with higher frequency and repeat the calculation of this step until satisfied.
2) For each variable X_i, obtain the fuzzy rules and membership functions on sample set $\tilde{B}_i$ by training

$$\mathcal{J}_{p,q}=\sum_{j=1}^N\prod_{i=1}^m\left(\cos\left(\frac{2\pi s_{p_{2i-1}}}{U_i-L_i}\left(X_{i,j}-\frac{U_i-L_i}{2}\right)+s_{p_{2i}}\cdot\frac{\pi}{2}\right)\cos\left(\frac{2\pi s_{q_{2i-1}}}{U_i-L_i}\left(X_{i,j}-\frac{U_i-L_i}{2}\right)+s_{q_{2i}}\cdot\frac{\pi}{2}\right)\right)$$

and

$$\mathcal{T}^k_{p,q}=\begin{cases}\sum_{j=1}^N Y_j\prod_{i=1}^m\cos\left(\frac{2\pi s_{k_{2i-1}}}{U_i-L_i}\cdot\left(X_{i,j}-\frac{U_i-L_i}{2}\right)+s_{k_{2i}}\cdot\frac{\pi}{2}\right) & \text{if } q=k\\ \sum_{j=1}^N\prod_{i=1}^m\Big(\cos\left(\frac{2\pi s_{p_{2i-1}}}{U_i-L_i}\cdot\left(X_{i,j}-\frac{U_i-L_i}{2}\right)+s_{p_{2i}}\cdot\frac{\pi}{2}\right) & \\ \quad\cdot\cos\left(\frac{2\pi s_{q_{2i-1}}}{U_i-L_i}\cdot\left(X_{i,j}-\frac{U_i-L_i}{2}\right)+s_{q_{2i}}\cdot\frac{\pi}{2}\right)\Big) & \text{if } q\ne k\end{cases}$$

the neural network in Fig. 1, where we choose $\tilde{B}_i = \{(X_{i,j}, \{f_{\bar{s},i,j} \,|\, (s_1, s_2, \ldots s_{2m}) = \bar{s} \in \mathcal{S}\} \,|\, j = 1, 2, \ldots, N\}$, The terminating condition of training each network is set as: $\max_{j \in \{1,2,\ldots,N\}} |O_{\mathrm{fuzzy}}(X_{i,j}) - f_{\bar{s},i,j}| \leq \varepsilon_{s1} = \min\{(1/m), (\varepsilon_1 - \varepsilon_{f1})/(mQ|\mathcal{S}|)\}$ and $\sum_{j=1}^{N} |O_{\mathrm{fuzzy}}(X_{i,j}) - f_{\bar{s},i,j}| \leq \varepsilon_{s2} = (\varepsilon_2 - \varepsilon_{f2})/(mQ|\mathcal{S}|(1 + (1/2)m\varepsilon_{s1}))$ for each output of the networks, where $Q = \max\{|a_{\bar{s}}|\}$.

3) Accumulate the fuzzy rules and the membership functions into an integrated fuzzy system on sample set Λ according to Theorem 2.

Theorem 2 also ensures that the maximum error and the sum of errors of the resulting fuzzy system on all of the samples are bounded by ε_1 and ε_2, respectively.

IV. Experimental Result

To test our computational scheme, we have tried approximating the same nonlinear function of eight variables $x_1, x_2, \ldots, x_8$, which we have used in ParNeuFuz and PolyNeuFuz [3], as given by

$$f(x_1, x_2, \ldots, x_8) = \sin(x_1) \times \cos(x_2) + (x_3)^3 \times \tan(x_4) - \ln(x_5 + 2)/(1 + (x_6)^2) + \sqrt{2x_7 \times x_8}. \quad (5)$$

We have randomly chosen 500 sampling points each from the following eight variable intervals: $-2 \leq x_1, x_2, x_3, x_4, x_6 \leq +2$, $-1 \leq x_5 \leq 3$, $0 \leq x_7, x_8 \leq 3$ so that 500 input-output data are used as the sample set. The error thresholds are set as $\epsilon_1 = 0.1$ and $\epsilon_2 = 25.0$, and initially we set $S = \{s_1, s_2, \ldots, s_{16} \,|\, s_1, s_3, \ldots, s_{15} \in \{0, 1, 2, 3\}, s_2, s_4, \ldots, s_{16} \in \{0, 1\}\}$, dividing the intervals of each variables $x_1, x_2, \ldots, x_8$ into three equally partitioned fuzzy sets with Gaussian membership functions each of which have a standard deviation of 0.5.

Using a Pentium 90 MHz computer in numerical experiments, we have obtained the final memberships and fuzzy rules of a fuzzy system represented by (5) within about 44 min with the final maximum error of $\epsilon_1 = 0.094$ and total error of $\epsilon_2 = 18.3$. The results of approximations are shown in Fig. 2. The solid lines show the results of function $f(x_1, x_2, \ldots, x_8)$ while the dotted lines show the approximations achieved by the fuzzy system. The 44 min of computing time by 1 CPU should be compared with the computing time of 3 h required of PolyNeuFuz by one CPU and 25 min of its parallel version ParNeuFuz by eight CPUs [3]. Although it is also possible for ParNeuFuz and PolyNewFuz to attain the similar final maximum error and sum of errors of the present FouNewFuz, the final results obtained by ParNeuFuz and by a single CPU version of PolyNewFuz has the final $\epsilon_1 = 0.089$ and $\epsilon_2 = 23.6$. However, as we mentioned in the introduction, the error bounds cited came from a trial and error basis and we cannot predict nor prespecify the final accuracy beforehand.

This is impressive in part because the computation of multivariable fuzzy systems exceeding eight variables is beyond the capability of most of the conventional algorithms, and also because the specified accuracy of computations can be implemented with much less computational resources needed in the preceding schemes of PolyNeuFuz [4], and its parallel version ParNeuFuz [3].

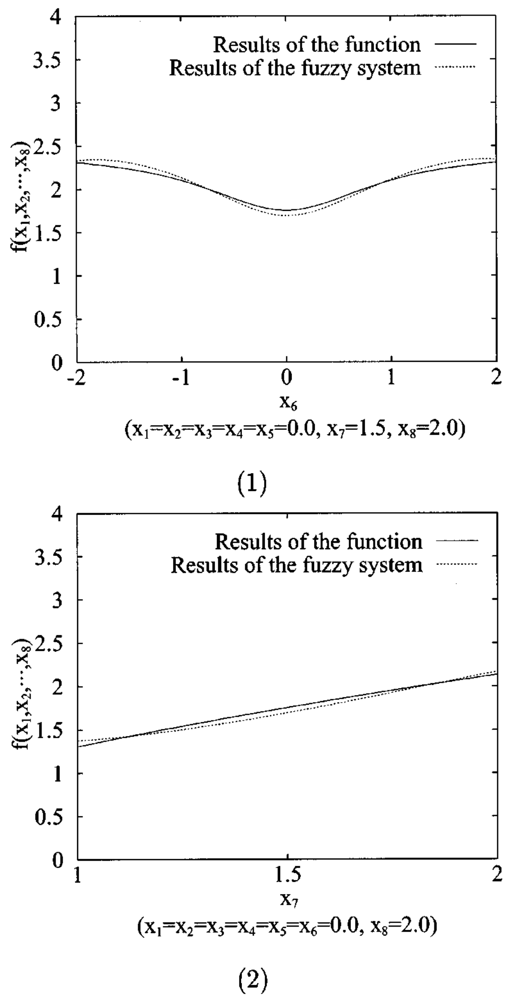

Fig. 2. Numerical simulation of 8-variable $f(x_1, x_2, \ldots, x_8)$ by *FouNeuFuz*.

Unlike ParNewFuz and PolyNeuFuz algorithms which can specify the overall accuracy in the multistep processes only by trials and errors, our Fourier-based method is capable of specifying the consistent error bounds at each of the steps so that an "over accuracy" can be avoided consistently thus without wasting the computing time.

V. Conclusion

This paper presented a new scheme for generating multivariable fuzzy systems. We have proved mathematically that the task of generating a multivariable fuzzy system can be decomposed into simpler tasks of generating a single input fuzzy system. Because of the independence of the input variables in computing different single input fuzzy systems, it is easy to show that our scheme can also be implemented on a parallel computer system. Our approach can be implemented as a practical tool for generating a fuzzy system in the areas involving moderately large number of variables. It can generate fuzzy systems with multiple inputs much faster than PolyNeuFuz [4], and perhaps ParNeuFuz [3], if we implement the algorithm with a multi-CPU system. This becomes possible by taking full advantage of an approach whereby we can control the error threshold values at any values. This is in sharp contrast to PolyNeuFuz and ParNeuFuz algorithms. As far as we know, no other methods are known to be capable of dealing with the same problem if the number of variables exceed five.

APPENDIX

A. Proof of Lemma 1

The assumption that fuzzy system $\mathrm{FS}_{\bar{s},i}(i = 1, 2, \ldots, m)$ can describe the sample set $\mathcal{B}_{\bar{s},i} = \{(X_{i,j}, f_{\bar{s},i,j}) \,|\, j = 1, 2, \ldots, N\}$ implies

$$f_{\bar{s},i,j} = \sum_{t_i=1}^{\tau_i} \mu_{i,t_i}(X_{i,j}) \cdot O_{\bar{s},i,t_i} + \varepsilon_{i,j}$$

where $|\varepsilon_{i,j}| \le \varepsilon_1$ and $\sum_{j=1}^{N} |\varepsilon_{i,j}| \le \varepsilon_2$.

By noting that for each $i = 1, 2, \ldots, m$,

$$\prod_{\bar{i}\neq i}\left(\sum_{t_{\bar{i}}=1}^{\tau_{\bar{i}}} \mu_{\bar{i},t_{\bar{i}}}(X_{\bar{i},j}) \cdot O_{\bar{s},\bar{i},t_{\bar{i}}} + \varepsilon_{\bar{i},j}\right) \cdot \varepsilon_{i,j} = \prod_{\bar{i}\neq i} f_{\bar{s},\bar{i},j} \cdot \varepsilon_{i,j}$$

and for (i_1, i_2) $(1 \le i_1 \neq i_2 \le m)$

$$\prod_{\bar{i}\neq i_1, \bar{i}\neq i_2}\left(\sum_{t_{\bar{i}}=1}^{\tau_{\bar{i}}} \mu_{\bar{i},t_{\bar{i}}}(X_{\bar{i},j}) \cdot O_{\bar{s},\bar{i},t_{\bar{i}}} + \varepsilon_{\bar{i},j}\right) \cdot \varepsilon_{i_1,j}\varepsilon_{i_2,j} = \prod_{\bar{i}\neq i_1, \bar{i}\neq i_2} f_{\bar{s},\bar{i},j} \cdot \varepsilon_{i_1,j}\varepsilon_{i_2,j}$$

we have

$$\begin{aligned} & f_{\bar{s},1,j} \cdot f_{\bar{s},2,j} \cdots f_{\bar{s},m,j} \\ & \quad = \prod_{i=1}^{m}\left(\sum_{t_i=1}^{\tau_i} \mu_{i,t_i}(X_{i,j})\right) \cdot \prod_{l=1}^{m} O_{\bar{s},l,t_l} \\ & \qquad + \sum_{i=1}^{m}\left(\varepsilon_{i,j}\prod_{\bar{i}\neq i} f_{\bar{i},j}\right) + O\left(\sum_{i_1<i_2} |\varepsilon_{i_1,j}\varepsilon_{i_2,j}|\right) \\ & \quad = \sum_{\substack{t_i=1,2,\ldots,\tau_i \\ i=1,2,\ldots,m}}\left(\prod_{l=1}^{m} \mu_{l,t_l}(X_{l,j})\right) \cdot O_{t_1,t_2,\ldots,t_m}^{(\bar{s})} \\ & \qquad + \sum_{i=1}^{m}\left(\varepsilon_{i,j}\prod_{\bar{i}\neq i} f_{\bar{i},j}\right) + O\left(\sum_{i_1<i_2} |\varepsilon_{i_1,j}\varepsilon_{i_2,j}|\right). \end{aligned}$$

Because of the fact that $|f_{\bar{s},i,j}| < 1$ for all $i = 1, 2, \ldots, m$ and $\sum_{i_1<i_2} |\varepsilon_{i_1,j}\varepsilon_{i_2,j}| < (1/2)(m\varepsilon_1)^2$, we have

$$\begin{aligned} & |O_{\text{fuzzy}}(\bar{X}_j) - f_{\bar{s},1,j} \cdot f_{\bar{s},2,j} \cdots f_{\bar{s},m,j}| \\ & \quad \le m\varepsilon_1 + O\left(\frac{1}{2}(m\varepsilon_1)^2\right) \end{aligned}$$

$$\begin{aligned} & \sum_{j=1}^{N} |O_{\text{fuzzy}}(\bar{X}_j) - f_{\bar{s},1,j} \cdot f_{\bar{s},2,j} \cdots f_{\bar{s},m,j}| \\ & \quad \le \sum_{i=1}^{m}\sum_{j=1}^{N} |\varepsilon_{i,j}| + \sum_{i_1<i_2}\left(\max|\varepsilon_{i_1,j}| \cdot \sum_{j=1}^{N} |\varepsilon_{i_2,j}|\right) \\ & \qquad + O\left(\sum_{i_1<i_2<i_3} |\varepsilon_{i_1,j}\varepsilon_{i_2,j}\varepsilon_{i_3,j}|\right). \end{aligned}$$

Hence, the proof is completed.

B. Proof of Lemma 2

The assumption that fuzzy system $\mathrm{FS}_{\bar{s}}(\bar{s} \in \mathcal{S})$ can be used to describe the sample set $\mathcal{C}_{\bar{s}}$ implies

$$g_{\bar{s}j} = \sum_{\substack{t_i=1,2,\ldots,\tau_i \\ i=1,2,\ldots,m}}\left(\prod_{l=1}^{m} \mu_{l,t_l}(X_{l,j}) \cdot O_{t_1,t_2,\ldots,t_m}^{(\bar{s})}\right) + \varepsilon_{\bar{s}j}$$

where $|\varepsilon_{\bar{s}j}| \le \varepsilon_1'$ and $\sum_{j=1}^{N} |\varepsilon_{\bar{s}j}| \le \varepsilon_2'$.

Thus, we readily have the following conclusion:

$$\begin{aligned} & \sum_{\bar{s}\in\mathcal{S}} (a_{\bar{s}} \cdot g_{\bar{s}j}) \\ & \quad = \sum_{\bar{s}\in\mathcal{S}}\left(\sum_{\substack{t_i=1,2,\ldots,\tau_i \\ i=1,2,\ldots,m}}\left(\prod_{l=1}^{m} \mu_{l,t_l}(X_{l,j}) \cdot O_{t_1,t_2,\ldots,t_m}^{(\bar{s})}\right)\right) \\ & \qquad + \sum_{\bar{s}} a_{\bar{s}}\varepsilon_{\bar{s}j} \\ & \quad = \sum_{\substack{t_i=1,2,\ldots,\tau_i \\ i=1,2,\ldots,m}}\left(\prod_{l=1}^{m} \mu_{l,t_l}(X_{l,j}) \cdot \sum_{\bar{s}\in\mathcal{S}}\left(a_{\bar{s}} \cdot O_{t_1,t_2,\ldots,t_m}^{(\bar{s})}\right)\right) \\ & \qquad + \sum_{\bar{s}} a_{\bar{s}}\varepsilon_{\bar{s}j} \\ & \quad = \sum_{\substack{t_i=1,2,\ldots,\tau_i \\ i=1,2,\ldots,m}}\left(\prod_{l=1}^{m} \mu_{l,t_l}(X_{l,j}) \cdot O_{t_1,t_2,\ldots,t_m}\right) \\ & \qquad + \sum_{\bar{s}} a_{\bar{s}}\varepsilon_{\bar{s}j} \\ & \quad = O_{\text{fuzzy}}(\bar{X}_j) + \sum_{\bar{s}} a_{\bar{s}}\varepsilon_{\bar{s}j}. \end{aligned}$$

Then, we have

$$\left|O_{\text{fuzzy}}(\bar{X}_j) - \sum_{\bar{s}\in\mathcal{S}} (a_{\bar{s}} \cdot g_{\bar{s}j})\right| \le \max_{\bar{s}\in\mathcal{S}}\}|a_{\bar{s}}||\mathcal{S}|\varepsilon_1'$$

$$\begin{aligned} & \sum_{j=1}^{N}\left|O_{\text{fuzzy}}(\bar{X}_j) - \sum_{\bar{s}\in\mathcal{S}} (a_{\bar{s}} \cdot g_{\bar{s}j})\right| \\ & \quad \le \sum_{\bar{s}} \Bigg(\max_{\bar{s}\in\mathcal{S}} \Bigg\{ |a_{\bar{s}}| \sum_{j=1}^{N} \varepsilon_{\bar{s}j} \Bigg) \le |\mathcal{S}| \cdot \max_{\bar{s}\in\mathcal{S}} \Bigg\} \cdot \varepsilon_2'. \end{aligned}$$

Thus, the lemma is proved.

ACKNOWLEDGMENT

The authors wish to thank the referees for their constructive and valuable comments.

REFERENCES

[1] B. Kosko, “Fuzzy systems as universal approximator,” *IEEE Trans. Comput.*, vol. 43, pp. 1329–1333, 1994.
[2] A. Cauchy, “Methode generale pour la resolution des systemes d’equations simultanees,” in *Comp. Rend. Acad. Sci. Paris*, 1847, pp. 536–538.

[3] L. Chen, N. Tokuda, X. Zhang, and Y. He, "A new scheme for an automatic generation of multivariable fuzzy systems," *Fuzzy Sets Syst.*, vol. 120, pp. 323–329, 2001.

[4] L. Chen, J. Yan, and Y. He, "A new approach for the automatic generation of membership functions and rules of multivariable fuzzy system," *Proc. IEEE 1995 ICNN*, pp. 1342–1346, 1995.

[5] ——, "Automatic generation of fuzzy set membership functions & rules using a neural network," in *Proc. 3rd Australia New Zealand Intelligent Information Systems*, Perth, Australia, 1995, pp. 164–168.

[6] ——, "Automatic generating of fuzzy system by neural networks," *Chin. J. Acta Electron. Sinica*, vol. 24, no. 11, pp. 32–36, Nov. 1996. in Chinese.

[7] ——, "Using neural networks to extract fuzzy rules and membership functions," in *Proc. ICONIP*, vol. 2, Beijing, China, Oct. 1995, pp. 517–520.

[8] ——, "Generate multivariable fuzzy system through neural networks," *Proc. IEEE ICNNSP*, vol. 1, pp. 301–304, Dec. 1995.

[9] C. Czarnecki, R. John, and S. Bennett, "The application of fuzzy logic to real time multiple robot collision avoidance," in *Proc. ICSC Fuzzy Logic Symp.*, 1995, pp. C116–C121.

[10] R. D. Harding, *Fourier Series and Transforms*, Bristol, U.K.: Institute of Physics, 1985.

[11] H. Ishibuchi and K. Morioka, "Determination of type II membership functions by fuzzified neural networks," in *Proc. EUFIT*, 1995, pp. 529–533.

[12] H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed representation of fuzzy rules and its application to pattern classification," *Fuzzy Sets Syst.*, vol. 52, pp. 21–32, 1992.

[13] R. Jager, "Fuzzy logic in control," Ph.D. dissertation, Technische Universiteit Delft, Delft, The Netherlands, 1995.

[14] J.-S. R. Jang, "Self learning fuzzy controllers based on temproal back propagation," *IEEE Trans. Neural Networks*, vol. 3, pp. 714–723, 1992.

[15] C. Karr, "Design of an adaptive fuzzy logic controller using a genetic algorithm," in *Proc. 4th ICGA*, 1991, pp. 450–457.

[16] N. K. Kasabov, "Learning fuzzy rules through neural networks," in *Proc. 1st First New Zealand Int. Conf. Artificial Neural Networks and Expert Systems*, 1993, pp. 137–139.

[17] E. Khan, "Neural network-based algorithms for rule evaluation and defuzzification in fuzzy logic design," in *Proc. WCNN*, vol. II, Portland, OR, 1993, pp. 31–38.

[18] E. Khan and F. Unal, "Recurrent fuzzy logic using neural network," in *Lecture Notes in Computer Science*. New York: Springer-Verlag, Aug. 1995, vol. 1011, pp. 48–55.

[19] E. Khan and P. Venkatapuram, "Neufuz: Neural network-based fuzzy logic design algorithms," *Proc. IEEE 1993 ICFS*, pp. 647–654, 1993.

[20] C. C. Lee, "Fuzzy Logic in control systems—Part II: Fuzzy logic controller," *IEEE Trans. Syst, Man, Cybern.*, vol. 20, pp. 419–435, 1990.

[21] D. G. Luenberger, *Linear and Nonlinear Programming*, 2nd ed. Reading, MA: Addison-Wesley, 1984.

[22] D. L. Meredith, C. L. Karr, and K. K. Kamur, "The use of genetic algorithms in the design of fuzzy logic controllers," in *Proc. 3rd Workshop Neural Networks*, 1992, pp. 549–545.

[23] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numberical Recipes*. Cambridge, U.K.: Cambridge Univ. Press, 1986.

[24] T. J. Schwartz, "Fuzzy systems in the real world," *AI Expert*, vol. 5, pp. 28–36, 1990.

Liang Chen, **[AUTHOR: Please E-MAIL text version of biography. Photo: 220 dpi, TIF, and black and white.]**

Naoyuki Tokuda (M'93), **[AUTHOR: Please E-MAIL text version of biography. Photo: 220 dpi, TIF, and black and white.]**