

A Functional-Link based Interval Type-2 Compensatory Fuzzy Neural Network for Nonlinear System modeling

Jyh-Yeong Chang

Department of Electrical
and Control Engineering,
National Chiao-Tung
University
Hsin-Chu, Taiwan, R.O.C.
jychang@mail.nctu.edu.tw

Yang-Yin Lin

Department of Electrical
and Control Engineering,
National Chiao-Tung
University
Hsin-Chu, Taiwan, R.O.C.
oliver.yylin@gmail.com

Ming-Feng Han

Department of Electrical
and Control Engineering,
National Chiao-Tung
University
Hsin-Chu, Taiwan, R.O.C.
ming0901@gmail.com

Chin-Teng Lin

Department of Electrical
and Control Engineering,
National Chiao-Tung
University
Hsin-Chu, Taiwan, R.O.C.
ctlin@mail.nctu.edu.tw

Abstract—In this paper, the Functional-Link based Interval Type-2 Compensatory Fuzzy Neural Network (FLIT2CFNN) is a six –layer structure, which combines compensatory fuzzy reasoning method, and the consequent part is combined the proposed functional-link neural network with interval weights. The compensatory fuzzy reasoning method uses adaptive fuzzy operations of neuro-fuzzy systems that can make the fuzzy logic system more adaptive and effective. Initially, there is no rule in the FLIT2CFNN. A FLIT2CFNN is constructed using concurrent structure and parameter learning. The advantages of this learning algorithm are that it converges quickly and the obtained fuzzy rules are more precise. All of the antecedent part parameters and compensatory degree values are learned by gradient descent algorithm. Several simulation results show that the FLIT2CFNN achieves better performance than other feedforward type-1 and type-2 FNNs.

Keywords- type-2 fuzzy systems, Compensatory, structure learning, on-line fuzzy clustering.

I. INTRODUCTION

Fuzzy neural networks have been successful in a variety of area. However, most of fuzzy neural networks handle nonlinear system or time-varying modeling that its popular issue of research. In recent years, studies on dealing with optimize fuzzy membership function and defuzzification scheme have drawn attention [1-3], and adjust the parameter of fuzzy membership functions and defuzzification functions to adaptive network. Unfortunately, for optimize fuzzy logic reasoning and select optimal fuzzy operators, only static fuzzy operators are often used to make fuzzy reasoning. Such that the conventional neural fuzzy system can only adjust fuzzy membership functions by using fixed fuzzy operations such as Min and Max, the compensatory neural fuzzy system [4] with adaptive fuzzy reasoning is more effective and adaptive than the conventional neural fuzzy system with non-adaptive fuzzy reasoning.

In the typical TSK-type neural fuzzy network (TSK-type NFN), which is a linear polynomial of input variables. Nevertheless, the traditional TSK-type neural fuzzy network does not take full advantage of the mapping capabilities that may be offered by the consequent part. However, the multilayer neural network has such disadvantages as slower convergence and greater computational complexity. Therefore, this studied uses the functional link neural network (FLNN) [5-6] to the consequent part of the fuzzy rules. The consequent part of the proposed FLNN model is a nonlinear combination of input variables, which differs from the other existing models [1-3,7]. Moreover, using the functional expansion can effectively increase the dimensionality of the input vector, so the hyper-planes generated by the FLNN will provide a good discrimination capability in input data space.

All of the aforementioned FNNs only use type-1 fuzzy sets. Recently, studies on type-2 FLSs have paid close attention [8,9] and a complete theory of type-2 have been developed in [10]. Type-2 FLS are extensions of type-1 FLS, where the membership value of a type-2 fuzzy set is a type-1 fuzzy number. The type-1 fuzzy set cannot directly deal with these uncertainties. Because the membership functions of type-2 fuzzy systems are themselves fuzzy, they supply a powerful framework to represent and handle such types of uncertainties. Successful applications of type-2 FLS include area of signal processes [11] and control [12]. Therefore, the purpose of this paper is to develop a self –evolving type-2 FNN with compensatory-based fuzzy logic reasoning and functional-link-based in the consequent part. The network is called a Functional-Link based Interval Type-2 Compensatory Fuzzy Neural Network (FLIT2CFNN).

The rest of this paper is organized as follows. Section II describes the FLIT2CFNN structure. Section III introduces structure and parameter learning in a FLIT2CFNN. Section IV simulates nonlinear system identification problems. Finally, Section V draws conclusions.

II. FLIT2CFNN STRUCTURE

In this section, the structure of Functional-Link based Interval Type-2 Compensatory Fuzzy Neural Network (FLIT2CFNN) is introduced. Figure 1 shows the structure of FLIT2CFNN, which has a six-layer. This six-layer network realizes an interval type-2 fuzzy system whose consequent part is a nonlinear combination of functional expansion of input variables, i.e., functional-link based. The detailed mathematical functions of FLIT2CFNN are introduced as following below.

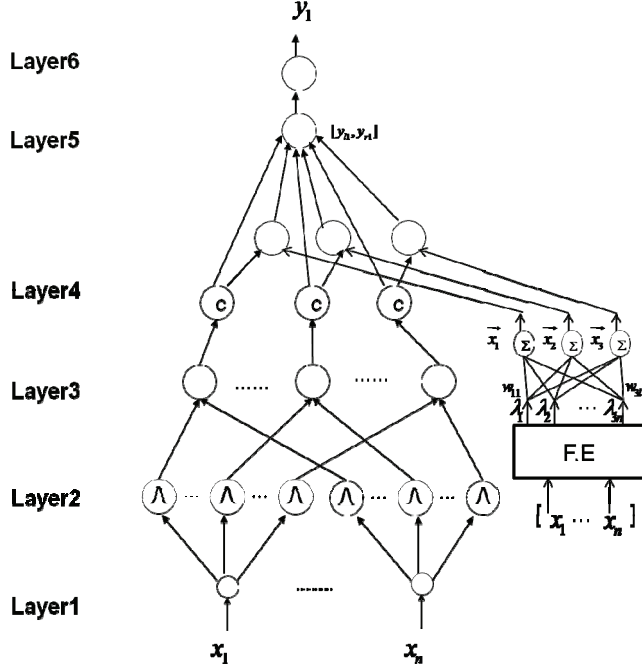


Fig.1. Structure of the FLIT2CFNN

Layer 1 (input layer): The inputs are crisp values. Only the current states $x(t)$ are fed as inputs to this layer. Note that there is no weights to be adjusted in this layer.

Layer 2 (MF Layer): Each node in this layer defines an interval type-2 membership function (MF). We use a Gaussian primary MF having a fixed standard deviation (STD) and an uncertain mean that takes on values in $[m_1, m_2]$ (Fig. 2), i.e., That is, the output $\tilde{A}_j^i$ of each node can be represented as an interval $[\underline{\mu}_{\tilde{A}_j^i}, \bar{\mu}_{\tilde{A}_j^i}]$.

$$\mu_{\tilde{A}_j^i} = \exp\left\{-\frac{1}{2}\left(\frac{x_j - m_j^i}{\sigma_j^i}\right)^2\right\} \equiv N(m_j^i, \sigma_j^i; x_j), \quad m_j^i \in [m_{j1}^i, m_{j2}^i] \quad (1)$$

The footprint of uncertainty (FOU) of this MF can be represented as a bounded interval in terms of upper MF, $\bar{\mu}_{\tilde{A}_j^i}$, and lower MF, $\underline{\mu}_{\tilde{A}_j^i}$,

where

$$\bar{\mu}_{\tilde{A}_j^i}(x_j) = \begin{cases} N(m_{j1}^i, \sigma_j^i; x_j), & x_j < m_{j1}^i \\ 1, & m_{j1}^i \leq x_j \leq m_{j2}^i \\ N(m_{j2}^i, \sigma_j^i; x_j), & x_j > m_{j2}^i \end{cases} \quad (2)$$

and

$$\underline{\mu}_{\tilde{A}_j^i}(x_j) = \begin{cases} N(m_{j2}^i, \sigma_j^i; x_j), & x_j \leq \frac{m_{j1}^i + m_{j2}^i}{2} \\ N(m_{j1}^i, \sigma_j^i; x_j), & x_j > \frac{m_{j1}^i + m_{j2}^i}{2} \end{cases} \quad (3)$$

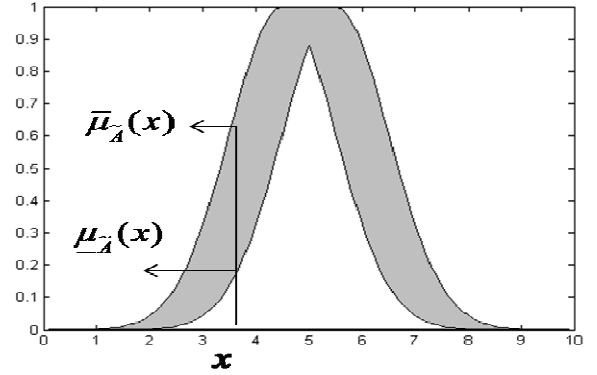


Fig.2. An interval type-2 fuzzy set with an uncertain mean.

Layer 3 (Spatial Firing layer): Each node performs a fuzzy meet operation on inputs from layer 2 using an algebraic product operation to obtain the spatial firing strength. The output of a rule node is a firing strength, F^i , which is an interval type-1 fuzzy set. The firing strength is computed as follows (Liang and Mendel, 2000)

$$F^i = [\bar{f}^i, \underline{f}^i] \quad i = 1 \sim M, \quad (4)$$

where

$$\bar{f}^i = \prod_{j=1}^n \bar{\mu}_{\tilde{A}_j^i} \quad \text{and} \quad \underline{f}^i = \prod_{j=1}^n \underline{\mu}_{\tilde{A}_j^i} \quad (5)$$

and M is total number of rules.

Layer 4 (Compensatory Firing layer): Each node in this layer performs a compensatory fuzzy operation mentioned in [4], and derives if-condition matching of fuzzy rules. As a result, the compensatory firing function of inference node is

$$\phi^i = [\bar{\phi}^i, \underline{\phi}^i], \quad i = 1, \dots, M \quad (6)$$

where

$$\bar{\phi}^i = \left(\prod_j \bar{f}_j^i \right)^{(1-\gamma_i + \frac{\gamma_i}{n})}, \quad \underline{\phi}^i = \left(\prod_j \underline{f}_j^i \right)^{(1-\gamma_i + \frac{\gamma_i}{n})} \quad (7)$$

Here, the $\gamma_i \in [0, 1]$ is called compensatory degree. The purpose of tuning γ_i is to increase the adaptability of fuzzy operation.

Layer 5(Consequent and Output processing layer): Each rule node in layer 3 has its own corresponding consequent node in layer 4. The output of each node is an interval type-1 fuzzy set, denoted by $[w_l^i, w_r^i]$, if function-link neural network is used, then the link is existent.

$$[w_l^i, w_r^i] = [c_0^i - s_0^i, c_0^i + s_0^i] + \sum_{k=1}^L [c_k^i - s_k^i, c_k^i + s_k^i] \cdot \lambda_k \quad (8)$$

That is,

$$w_l^i = \sum_{k=0}^L c_k^i \lambda_k - \sum_{k=0}^L |\lambda_k| s_k^i \quad (9)$$

where $\lambda_0 = 1$, and $L=3n$

$$w_r^i = \sum_{k=0}^L c_k^i \lambda_k + \sum_{k=0}^L |\lambda_k| s_k^i \quad (10)$$

Here λ_k is the functional expansion of input variables. The functional expansion uses a trigonometric polynomial basis function, given by $[x_1 \sin(\pi x_1) \cos(\pi x_1) x_2 \sin(\pi x_2) \cos(\pi x_2)]$ for two-dimensional input variables.

For output processing, the output y_l and y_r can be computed by using KM procedure [10]. In that procedure, the consequent part weights should be re-arranged in ascending order. According to [13], the relationship between $y_l, y_r, \tilde{y}_l$, and $\tilde{y}_r$ is

$$y_l = Q_l \tilde{y}_l \text{ and } y_r = Q_r \tilde{y}_r \quad (11)$$

where Q_l and Q_r are $M \times M$ permutation matrices. Re-order the f^i accordingly and call them g^i . The outputs y_l and y_r can be computed as follows

$$y_l' = \frac{\sum_{i=1}^L \bar{g}^i y_l^i + \sum_{i=L+1}^M \underline{g}^i y_l^i}{\sum_{i=1}^L \bar{g}^i + \sum_{i=L+1}^M \underline{g}^i} \quad (12)$$

and

$$y_r' = \frac{\sum_{i=1}^R \underline{g}^i y_r^i + \sum_{i=R+1}^M \bar{g}^i y_r^i}{\sum_{i=1}^R \underline{g}^i + \sum_{i=R+1}^M \bar{g}^i} \quad (13)$$

Layer 6 (Output layer): The node in this layer computes the output linguistic variable y by defuzzification operation. The output of layer 5 is an interval set as a result of computing average of y_l' and y_r' . Hence, the defuzzified output is

$$y = \frac{y_l' + y_r'}{2} \quad (14)$$

III. FLIT2CFNN LEARNING

In this section, the FLIT2CFNN simultaneously uses on-line two types of learning algorithm, the structure and parameter learning. There are no rules in the FLIT2CFNN

initially. The rules are created dynamically and automatically as learning proceeds upon receiving on-line incoming training data by performing the structure and parameter learning processes. All of free parameters are learned by following learning as a rule is generated. The details of structure learning and parameter learning are introduced in the rest of this section.

A. Structure learning

The first task in structure learning is conducted to decide whether or not to extract a new rule from training data as well as the number of fuzzy sets on the universal of discourse of each input variable. A previous study [2] used the rule firing strength as a criterion for type-1 fuzzy rule generation. This idea is extended to type-2 fuzzy rule generation criteria in a FLIT2CFNN. The firing strength is an interval in the FLIT2CFNN so that the center of interval is computed as

$$f_c^i = \frac{1}{2}(\bar{f}^i + \underline{f}^i) \quad (15)$$

The firing strength center then serves as a rule generation criterion. That is, for each piece of incoming data $\bar{x} = (x_1, \dots, x_n)$, find

$$I = \arg \max_{1 \leq i \leq M(t)} f_c^i(\bar{x}) \quad (16)$$

where $M(t)$ is the number of existing rules at time t . If $f_c^I(\bar{x}) \leq \phi_{th}$, then a new rule is generated, where $\phi_{th} \in (0, 1)$ is a pre-specified threshold that decays during the learning process. The threshold decides the number of input clusters generated in FLIT2CFNN.

Once a new fuzzy set is generated, the initial uncertain means and widths of the corresponding new type-2 fuzzy sets are assigned as

According to the notation in Eq. (1), the initial uncertain mean m_j^i and standard deviation σ_j^i for the new interval type-2 fuzzy set in input variable x_j are

$$m_j^i \in [x_j - 0.1, x_j + 0.1] \quad (17)$$

$$\sigma_j^i = \sigma_{fix} \quad (18)$$

B. Parameter Learning

The parameter learning phase occurs concurrently with the structure learning phase. For each incoming data, all of free parameters in the FLIT2CFNN are tuned, whether a newly generated rules or formerly existent. When we consider the single output case for clarity, our goal to minimize the error function is defined as

$$E(t) = \frac{1}{2} [y(t) - y^d(t)]^2 = \frac{1}{2} e^2(t) \quad (19)$$

where $y(t)$ and $y_d(t)$ denote real and desired outputs, respectively. The Karnik-Mendel iterative procedure for computing y_l and y_r in Eqs. (12) and (13) has a premise that y_l^i and y_r^i are re-arranged in ascending order.

Since the FLIT2CFNN training process, the consequent part parameters change, and the corresponding rule orders change accordingly. It is necessary to know the specific site of antecedent and consequent parameters for updating the parameters. According to this mapping, (12) and (13) are expressed by $\tilde{y}_l$ and $\tilde{y}_r$, i.e., with the consequent values arranged in the original rule order despite their changes during the parameter learning process. Based on this original rule-ordered expression, the consequent part parameters are tuned using Kalman filter algorithm [14]. The compensatory degree weights and the antecedent part parameters are tuned using the gradient descent algorithm, and it is expressed as

$$w_i(t+1) = w_i(t) - \eta \frac{\partial E}{\partial w_i}, \quad (20)$$

where w can be expressed as $m_{j1}, m_{j2}, \sigma, \gamma$, and η is a learning coefficient. Details of the learning algorithm can be found in [13].

IV. SIMULATIONS

To demonstrate the ability of FLIT2CFNN to evolve, the following nonlinear system is described by the difference equation [1-2]

$$y_d(t+1) = \frac{y_d(t)}{1 + y_d^2(t)} + u^3(t) \quad (21)$$

The training patterns are generated with $u(t) = \sin(2\pi t/100)$, $t=1, \dots, 200$. The inputs of FLIT2CFNN are $y_d(t)$ and $u(t)$, and the desired output is $y_d(t+1)$. Performance is evaluated using root-mean-square-error (RMSE)

$$RMSE = \sqrt{\frac{1}{200} \sum_{k=1}^{200} [y(t+1) - y^d(t+1)]^2} \quad (22)$$

where $y(t)$ is the output of FLIT2CFNN. Learning coefficient η is set to be 0.075. During the training process, the three rules are generated. Training is performed for 500 iterations. Table I shows that the performance of the FLIT2CFNN contains the network size, training and test RMSEs. Figure 3 shows the outputs of the plant and the IT2RFNN for these test inputs. The performance of FLIT2CFNN is compared with that of type-1 and type-2 FNNs, including SONFIN[2] and Feedforward type-2 FNN, where the former parameter learning are learned by hybrid learning of Kalman filter algorithm and gradient descent, and the latter free parameters are tuned by gradient descent method. The results show that the FLIT2CFNN achieves smaller test errors than the other feedforward networks.

TABLE I. PERFORMANCE OF FLIT2CFNN AND OTHER FEEDFORWARD SYSTEM

Model	SONFIN	FEEDFORWARD TYPE-2 FNN	FLIT2CFNN
Rule number	10	6	3
Parameter number	70	72	66
Iterations	250	250	250
RMSE	0.0075	0.0082	0.0035

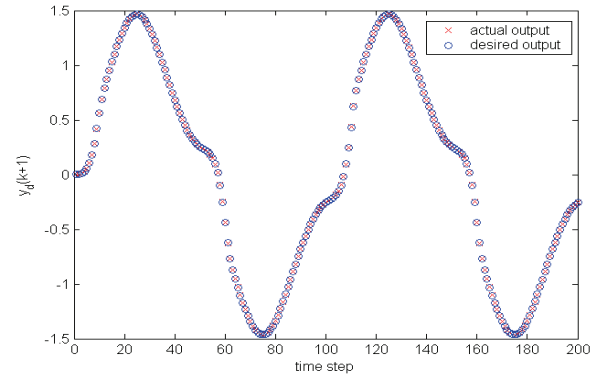


Fig.3. Output of the nonlinear system (O) and FLIT2CFNN (X).

This example is also considered the noise resistance ability of an FLIT2CFNN. Assumed that the measure y^d output contains noise. The added noise is artificially generated and is uniformly distributed in $[-2.5, 2.5]$. There are 20 Monte-Carlo realizations. On average, three rules are generated for the FLIT2CFNN. Fig. 4 shows one learning result of the 20 realizations, showing that the FLIT2CFNN output does not change as violently as the actual noisy output. This result verifies the FLIT2CFNN's noise resistance abilities. Table II shows the performance of FLIT2CFNN with noise. For comparison, the self-evolving type-1 system, SONFIN[2] and the Feedforward type-2 FNN are compared to the same problem. These results show that the FLIT2CFNN learning performance is better than the SONFIN and Feedforward type-2 FNN learning performance when there is noise.

TABLE II. PERFORMANCE OF FLIT2CFNN AND OTHER FEEDFORWARD SYSTEM WITH NOISE STATE

Model	SONFIN	FEEDFORWARD TYPE-2 FNN	FLIT2CFNN
Rule number	10	6	3
Parameter number	70	72	66
Train RMSE	0.13	0.142	0.128
Test RMSE	0.095	0.088	0.066

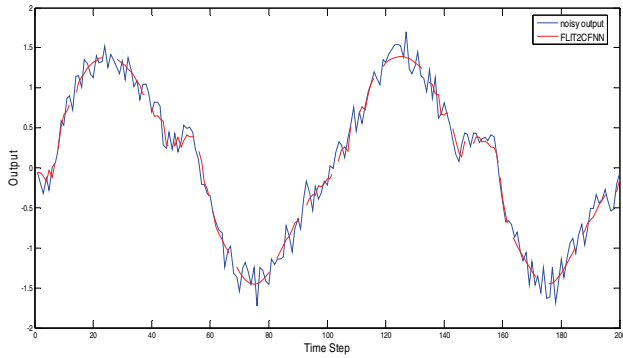


Fig.4. Online learning results of the FLIT2CFNN when noise appears in the measured data

V. CONCLUSIONS

This paper proposes a functional-link-based Interval Type-2 Compensatory Fuzzy Neural Network (FLIT2CFNN) structure for nonlinear system modeling and resistance noise ability. To achieve optimize type-2 fuzzy logic reasoning, the compensatory operation is chosen. The FLIT2CFNN model can automatically construct and adjust free parameters by performing online structure/parameter learning schemes concurrently. Moreover, the rule-ordered Kalman filter algorithm helps tune the consequent parameters online and improves learning accuracy. Except that the FoU in FLIT2CFNN helps handle the numerical uncertainty associated with system inputs and outputs. Therefore, the FLIT2CFNN has the potential to achieve better performance than type-1 fuzzy systems when dealing with noisy data. The proposed FLIT2CFNN can efficiently handle nonlinear system and good performance is achieved.

REFERENCES

- [1] C.J. Lin and C. T. Lin, "An ART-Based Fuzzy Adaptive Learning Control Network" *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 4, pp.477-496 Nov. 1997.
- [2] F. Juang and C. T. Lin, "An On-Line Self-Constructing Neural Fuzzy Inference Network and Its Applications," *IEEE Transactions on Fuzzy Systems*, vol. 6, no.1, pp. 12-31, Feb. 1998
- [3] F. J. Lin, C. H. Lin, and P. H. Shen, "Self-Constructing Fuzzy Neural Network Speed Controller for Permanent-Magnet Synchronous Motor Drive," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 5, pp. 751-759, Oct. 2001.
- [4] Y. Q. Zhang and A. Kandel, "Compensatory Neurofuzzy Systems with Fast Learning Algorithms," *IEEE Transactions on Neural Networks*, vol. 9, no. 1, pp. 83-105, Jan. 1998.
- [5] Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*, MA: Addison-Wesley, 1989.
- [6] J. C. Patra, R. N. Pal, B. N. Chatterji, and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks," *IEEE Trans. on Syst., Man, and Cybern.*, vol. 29, Apr. 1999.
- [7] T. Lin and C. F. Juang, "An adaptive neural fuzzy filter and its applications," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 27, no. 4, pp. 635-656, Aug. 1997.

- [8] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 6, pp. 643-658, Dec. 1999.
- [9] J. M. Mendel and R. I. John, "Type-2 fuzzy sets made simple," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 117-127, Apr. 2002.
- [10] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic System: Introduction and New Directions*. Upper Saddle River, NJ: Prentice-Hall, 2001.
- [11] Q. Liang and J. M. Mendel, "Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 551-563, Oct. 2000.
- [12] H. Hagras, "A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots," *IEEE Trans. Fuzzy Syst.*, vol. 12, no. 4, pp. 524-539, Aug. 2004.
- [13] J. M. Mendel, "Computing derivatives in interval type-2 fuzzy logic system," *IEEE Trans. On Fuzzy Systems*, vol. 12. no. 1, pp. 84-98, Feb.2004.
- [14] C.F. Juang and Y. W. Tsao, "A self-evolving interval type-2 fuzzy neural network with on-line structure and parameter learning," *IEEE Trans. Fuzzy Systems*, vol. 16, no. 6, pp. 1411-1424, Dec. 2008.