A Novel Recurrent Interval Type-2 Fuzzy Neural Network for Nonlinear Channel Equalization

Ching-Hung Lee, *Member, IAENG*, Hao-Han Chang, Che-Ting Kuo, Jen-Chieh Chien, and Tzu-Wei Hu

Abstract—**In this paper, we propose a novel recurrent interval type-2 fuzzy neural network with asymmetric membership functions (RT2FNN-A). The RT2FNN-A uses the interval asymmetric type-2 fuzzy sets and it implements the fuzzy logic system (FLS) in a five-layer neural network structure. The RT2FNN-A is modified from the type-2 fuzzy neural network to provide memory elements for capturing the system's dynamic information and has the properties of high approximation accuracy and small network structure. Based on the Lyapunov theorem and gradient descent method, the convergence of RT2FNN-A is guaranteed and the corresponding learning algorithm is derived. In addition, the RT2FNN-A is applied in the nonlinear channel equalization to show the performance and effectiveness of RT2FNN-A system.**

Index Terms—**type-2 fuzzy logic system, recurrent neural network, asymmetric membership functions, channel equalization**

I. Introduction

In recent years, the fuzzy systems and control are regarded as the most widely used application of fuzzy logic system [5-9, 22]. Mendel and Karnik developed a complete theory of type-2 fuzzy logic systems (T2FLSs) [8, 16, 21-22]. Recently, T2FLSs have attracted more attention in many literatures and special issues [1, 6, 10-11, 16, 21-22, 24].

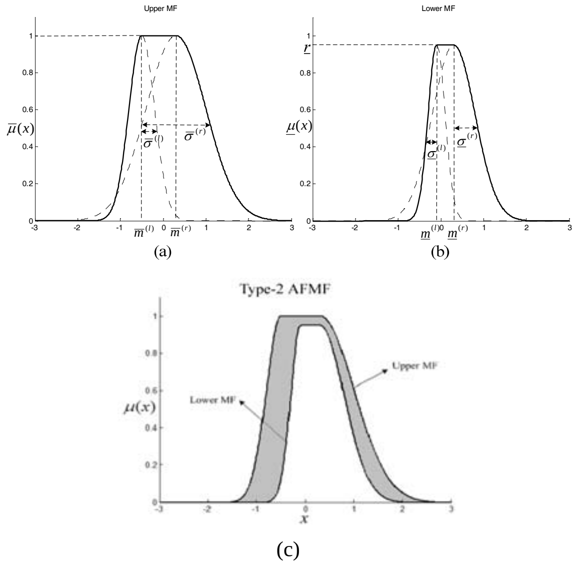
The major difference being the present of type-2 is their antecedent and consequent sets. T2FLSs result in better performance than type-1 fuzzy logic systems on the applications of function approximation, modeling, and control. Besides, neural networks have found numerous practical applications, especially in the areas of prediction, classification, and control [15]. Based on the advantages of T2FLSs and neural networks, the type-2 fuzzy neural network (T2FNN) systems are presented to handle the system uncertainty and reduce the rule number and computation [1, 10-11, 22, 24]. Using the asymmetric Gaussian function which is a new type of membership function due to excellent approximation results it provides. It also provides a fuzzy-neural network with higher flexibility to easily approach the optimum result more accurately. In the literature [9-10, 23], a T2FNN with asymmetric membership functions (T2FNN-A) was proposed to improve the system performance and obtain better approach ability.

In recent years, the nonlinear channel equalization using intelligent system was discussed [1-2, 17, 19-20]. Literature [4] introduced the ionospheric scintillation phenomenon which may cause multi-path problem when signal was transmitted. The multi-path problem is similar to nonlinear time-varying channel. In [19], there were successful application cases in complex channel equalization by using self-constructing fuzzy neural network, but a larger fuzzy rule number should be used when signal to noise ratio is low.

In this paper, we proposed a combining interval type-2 fuzzy asymmetric membership functions with recurrent neural network system, called RT2FNN-A. The proposed RT2FNN-A is a modified version of the T2FNN [11-14, 20, 24] which provides memoried elements to capture system dynamic response [15]. The RT2FNN-A system capability for temporarily storing information allowed us to extend the application domain to include temporal problem. Simulations are shown to illustrate the effectiveness of the RT2FNN-A system.

II. Interval type-2 fuzzy neural network systems

We first introduce the recurrent type-2 neural fuzzy inference system with asymmetric fuzzy MFs (RT2FNN-A) that was modified and extended from previous results [4, 5, 8, 12]. The RT2FNN-A uses the interval asymmetric type-2 fuzzy sets and it implements the FLS in a five-layer neural network structure which contains four-layer forward network and a feedback layer.



(a) (b)

(c)

Figure 1: Construction of a type-2 AFMF: (a) upper MF (solid line), (b) lower MF (solid line), and (c) constructed type-2 AFMF.

A. Construction of Type-2 Asymmetric Fuzzy Membership Functions

In general, given an system input data set x_i , i=1, 2, …, n, and the desired output y_p , p=1, 2, …, m, the jth type-2 fuzzy rule has the form

This work was supported in part by the National Science Council, Taiwan, R.O.C., under contracts NSC-95-2221-E-155-068-MY2.

Ching-Hung Lee is with Department of Electrical Engineering, Yuan-Ze University, Chung-li, Taoyuan 320, Taiwan. (phone: +886-3-4638800, ext: 7119; fax: +886-3-4639355, e-mail: chlee@saturn.yzu.edu.tw).

Rule j: IF x_1 is $\tilde{F}_1^j$ and … x_n is $\tilde{F}_n^j$
THEN y_1 is $\tilde{w}_1^j$ and … y_m is $\tilde{w}_m^j$

where j is the number of rules, $\tilde{F}_i^j$ represents the linguistic term of the antecedent part, $\tilde{w}_p^j$ represents the real number of the consequent part; n and m are the numbers of the input and output dimensions, respectively. Besides, the representation of jth rule for RT2FNN-A is

Rule j: IF x_1 is $\tilde{G}_{1j}$ and … x_n is $\tilde{G}_{nj}$ and g_j is $\tilde{G}_j^F$ THEN y_1 is $\tilde{w}_1^j$ and … y_m is $\tilde{w}_m^j$, g_1 is $\tilde{a}_1^j$, g_2 is $\tilde{a}_2^j$, …, and g_M is $\tilde{a}_M^j$.

where $\tilde{G}_{ij}$ represents the linguistic term of the antecedent part, $\tilde{w}_p^j$ and $\tilde{a}_p^j$ represents the interval real number of the consequent part; and M is the rule number. Here the fuzzy MFs of the antecedent part $\tilde{G}_{ij}$ are asymmetric interval type-2 fuzzy sets, which represent the different from typical Gaussian MFs.

Here we use the superscripts (l) and (r) to denote the left and right curves of a Gaussian MF. The parameters of lower and upper MFs are denoted by an underline (_) and bar (¯), respectively. Thus, the upper MF is constructed as

$$\bar{\mu}_{\tilde{F}}(x) = \begin{cases} \exp\left[-\frac{1}{2}\left(\frac{x-\bar{m}^{(l)}}{\bar{\sigma}^{(l)}}\right)^2\right], & \text{for } x \le \bar{m}^{(l)} \\ 1, & \text{for } \bar{m}^{(l)} \le x \le \bar{m}^{(r)} \\ \exp\left[-\frac{1}{2}\left(\frac{x-\bar{m}^{(r)}}{\bar{\sigma}^{(r)}}\right)^2\right], & \text{for } \bar{m}^{(r)} \le x \end{cases} \quad (1)$$

where $\bar{m}^{(l)}$ and $\bar{m}^{(r)}$ denote the means of two Gaussian MFs satisfying $\bar{m}^{(l)} \le \bar{m}^{(r)}$, and $\bar{\sigma}^{(l)}$ and $\bar{\sigma}^{(r)}$ denotes the deviation (i.e., width) of two Gaussian MFs. Figure 1(a) shows the upper type-2 AFMF constructed using $\bar{m}^{(l)}$, $\bar{m}^{(r)}$, $\bar{\sigma}^{(l)}$, and $\bar{\sigma}^{(r)}$. Similarly, the lower asymmetric MF is defined as

$$\underline{\mu}_{\tilde{F}}(x) = \begin{cases} \underline{r}\cdot\exp\left[-\frac{1}{2}\left(\frac{x-\underline{m}^{(l)}}{\underline{\sigma}^{(l)}}\right)^2\right], & \text{for } x \le \underline{m}^{(l)} \\ \underline{r}, & \text{for } \underline{m}^{(l)} \le x \le \underline{m}^{(r)} \\ \underline{r}\cdot\exp\left[-\frac{1}{2}\left(\frac{x-\underline{m}^{(r)}}{\underline{\sigma}^{(r)}}\right)^2\right], & \text{for } \underline{m}^{(r)} \le x \end{cases} \quad (2)$$

where $\underline{m}^{(l)} \le \underline{m}^{(r)}$ and $0.5 \le \underline{r} \le 1$. The corresponding widths of the MFs are $\underline{\sigma}^{(l)}$ and $\underline{\sigma}^{(r)}$. To avoid unreasonable MFs, the following constrains are added:

$$\begin{cases} \bar{m}^{(l)} \le \underline{m}^{(l)} \le \underline{m}^{(r)} \le \bar{m}^{(r)} \\ \underline{\sigma}^{(l)} \le \bar{\sigma}^{(l)} \;,\; \underline{\sigma}^{(r)} \le \bar{\sigma}^{(r)}. \\ 0.5 \le \underline{r} \le 1 \end{cases} \quad (3)$$

Figure 1(b) sketches the lower type-2 AFMF. The corresponding constructed type-2 AFMF is shown in Fig. 1(c). This introduces the properties of uncertain mean and variance [14]. Additionally, we can construct other type-2 asymmetric MFs by tuning the parameters. The corresponding tuning algorithm is derived to improve system accuracy and approximation ability.

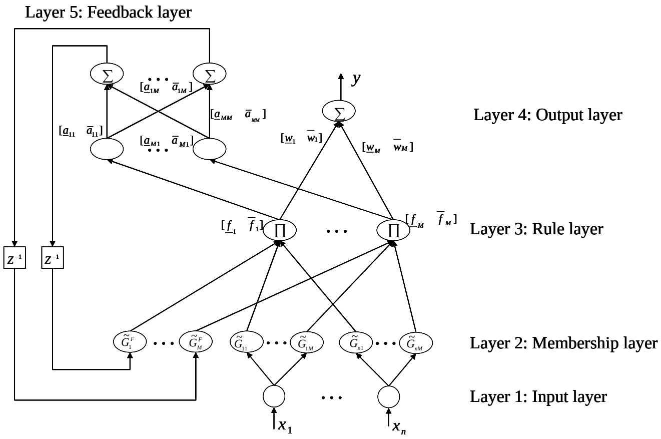

Layer 5: Feedback layer
Layer 4: Output layer
Layer 3: Rule layer
Layer 2: Membership layer
Layer 1: Input layer
Σ
Π
y
$[\underline{a}_{11} \;\; \bar{a}_{11}]$
$[\underline{a}_{1M} \;\; \bar{a}_{1M}]$
$[\underline{a}_{M1} \;\; \bar{a}_{M1}]$
$[\underline{a}_{MM} \;\; \bar{a}_{MM}]$
$[\underline{w}_1 \;\; \bar{w}_1]$
$[\underline{w}_M \;\; \bar{w}_M]$
$[\underline{f}_1 \;\; \bar{f}_1]$
$[\underline{f}_M \;\; \bar{f}_M]$
z^{-1}
z^{-1}
$\tilde{G}_1^F$
$\tilde{G}_M^F$
$\tilde{G}_{11}$
$\tilde{G}_{1M}$
$\tilde{G}_{n1}$
$\tilde{G}_{nM}$
x_1
x_n

Figure 2: Diagram of MISO recurrent type-2 fuzzy neural network with AFMFs (RT2FNN-A).

III. Structure of Recurrent Type-2 Fuzzy Neural Network With Asymmetric MFs (RT2FNN-A) System

The RT2FNN-A is shown in Fig. 2. In the following, we use $O_i^{(l)}$ to denote the ith output of a node in the lth layer.

Layer 1: Input Layer

For the ith node of layer 1, the net input and output are represented as

$$O_i^{(1)} = x_i^{(1)} \quad (4)$$

where $x_i^{(1)}$ represents the ith input to the jth node.

Layer 2: Membership Layer

In layer 2, it is clear that there are two parts in this layer, regular nodes and feedback nodes. Their input are $O_j^{(1)}$ and $g_j(k)$. Therefore, for network input x_i, the output is

$$O_{ij}^{(2)} = \left[\underline{O}_{ij}^{(2)} \quad \bar{O}_{ij}^{(2)}\right]^T = \left[\underline{\mu}_{\tilde{G}_{ij}}(O_i^{(1)}) \quad \bar{\mu}_{\tilde{G}_{ij}}(O_i^{(1)})\right]^T. \quad (5)$$

For internal or feedback variable g_j,

$$O_j^{F(2)} = \left[\underline{O}_j^{F(2)} \quad \bar{O}_j^{F(2)}\right]^T = \left[\underline{\mu}_{\tilde{G}_j^F}(g_j(k)) \quad \bar{\mu}_{\tilde{G}_j^F}(g_j(k))\right]^T \quad (6)$$

where the subscript $_{ij}$ indicates the jth term of the ith input $O_i^{(1)}$, where j=1,…, M. The superscript ^{F} indicates the feedback layer.

Layer 3: Rule Layer

Using the product t-norm, the firing strength associated with the jth rule is

$$\underline{f}^j = \underline{\mu}_{\tilde{F}_1^j}(x_1) * \cdots * \underline{\mu}_{\tilde{F}_n^j}(x_n) * \underline{\mu}_{\tilde{G}_j^{jF}}(\cdot) \quad (7)$$

$$\bar{f}^j = \bar{\mu}_{\tilde{F}_1^j}(x_1) * \cdots * \bar{\mu}_{\tilde{F}_n^j}(x_n) * \bar{\mu}_{\tilde{G}_j^{jF}}(\cdot) \quad (8)$$

where $\underline{\mu}_{\tilde{G}_j}(\cdot)$ and $\bar{\mu}_{\tilde{G}_j}(\cdot)$ are the lower and upper membership grades of $\mu_{\tilde{G}}(\cdot)$. Therefore, a simple PRODUCT operation is used. Then, for the jth input rule node

$$\begin{aligned} O_j^{(3)} &= \left[\underline{O}_j^{(3)} \quad \bar{O}_j^{(3)}\right]^T \\ &= \left[\underline{O}_j^{F(2)} \prod_{i=1}^{n} w_{ij}^{(3)} \cdot \underline{O}_{ij}^{(2)} \quad \bar{O}_j^{F(2)} \prod_{i=1}^{n} w_{ij}^{(3)} \cdot \bar{O}_{ij}^{(2)}\right]^T \end{aligned} \quad (9)$$

where the weights $w_{ij}^{(3)}$ are assumed to be unity.

Layer 4: Output Layer

Without loss of generality, the consequent part of interval T2FLS is $\tilde{w}_j = \left[\underline{w}_j \quad \overline{w}_j\right]^T$, $\underline{w}_j \le \overline{w}_j$. The vector notations $\underline{w} = [\underline{w}_1 \quad \cdots \quad \underline{w}_M]^T$ and $\overline{w} = [\overline{w}_1 \quad \cdots \quad \overline{w}_M]^T$ are used for clarity. According to the literature [23], we denote the maximum and minimum of $\sum_{i=1}^{M} f_i w_i$ as $\overline{O}^{(4)}$ and $\underline{O}^{(4)}$.

$$\underline{O}^{(4)} = \underline{w}^T f_L = \sum_{j=1}^{L}\left(\overline{O}_j^{(3)} \underline{w}_j\right) + \sum_{j=L+1}^{M}\left(\underline{O}_j^{(3)} \underline{w}_j\right), \tag{10}$$

$$\overline{O}^{(4)} = \overline{w}^T f_R = \sum_{j=1}^{R}\left(\underline{O}_j^{(3)} \overline{w}_j\right) + \sum_{j=R+1}^{M}\left(\overline{O}_j^{(3)} \overline{w}_j\right) \tag{11}$$

where

$$f_L = \left[\bar{f}_1, \cdots, \bar{f}_L, \underline{f}_{L+1}, \cdots, \underline{f}_M\right]^T = \left[\overline{O}_1^{(3)}, \cdots, \overline{O}_L^{(3)}, \underline{O}_{L+1}^{(3)}, \cdots, \underline{O}_M^{(3)}\right]^T \tag{12}$$

$$f_R = \left[\underline{f}_1, \cdots, \underline{f}_R, \bar{f}_{R+1}, \cdots, \bar{f}_M\right]^T = \left[\underline{O}_1^{(3)}, \cdots, \underline{O}_R^{(3)}, \overline{O}_{R+1}^{(3)}, \cdots, \overline{O}_M^{(3)}\right]^T . \tag{13}$$

It is obvious that *R* and *L* should be calculated first. The weights are arranged in order as $\underline{w}_1 \le \underline{w}_2 \le \cdots \underline{w}_M$ and $\overline{w}_1 \le \overline{w}_2 \le \cdots \overline{w}_M$. According to the Karnik-Mendel procedure [8, 16, 22], *L* and *R* are

$$L = \underset{j \in [1,\cdots,M-1]}{\arg} \min\left(\underline{O}^{(4)}\right), \quad R = \underset{j \in [1,\cdots,M-1]}{\arg} \max\left(\overline{O}^{(4)}\right). \tag{14}$$

Finally, the crisp output is

$$O^{(4)} = \frac{\underline{O}^{(4)} + \overline{O}^{(4)}}{2}. \tag{15}$$

Layer 5: Feedback Layer

This layer contains the context nodes which is used to produce the internal or feedback variable g_j. Each rule is associated with a particular internal variable. Hence, the number of the context nodes is equal to the number of rules. The same operations (type-reduction and defuzzification) as layer 4 are performed here.

$$\underline{O}_j^{(5)}(k+1) = \underline{a}_j^T f_L = \sum_{h=1}^{L_j^F}\left(\overline{O}_h^{(3)} \underline{a}_{jh}\right) + \sum_{h=L_j^F+1}^{M}\left(\underline{O}_h^{(3)} \underline{a}_{jh}\right) \tag{16}$$

$$\overline{O}_j^{(5)}(k+1) = \overline{a}_j^T f_R = \sum_{h=1}^{R_j^F}\left(\underline{O}_h^{(3)} \overline{a}_{jh}\right) + \sum_{h=R_j^F+1}^{M}\left(\overline{O}_h^{(3)} \overline{a}_{jh}\right) \tag{17}$$

and

$$L_j^F = \underset{h \in [1.\cdots,M-1]}{\arg} \min\left(\underline{O}_j^{(5)}(k+1)\right), \tag{18}$$

$$R_j^F = \underset{h \in [1.\cdots,M-1]}{\arg} \max\left(\overline{O}_j^{(5)}(k+1)\right). \tag{19}$$

Finally, the crisp output of this layer is

$$g_j(k+1) = O_j^{(5)}(k+1) = \frac{1}{2}\left[\underline{O}_j^{(5)}(k+1) + \overline{O}_j^{(5)}(k+1)\right] \tag{20}$$

Note that the delayed value of g_j is fed into layer 2, and it acts as an input variable to the precondition part of a rule. Each fuzzy rule has the corresponding internal variable g_j which is used to decide the influence degree of temporal history to the current rule.

A. Learning Algorithm for RT2FNN-A System

The gradient descent method is adopted to derive learning algorithm of the RT2FNN-A system. For clarification, we consider the single-output system and define the error cost function as

$$E(k) = \frac{1}{2}[y_d(k) - \hat{y}(k)]^2 \tag{21}$$

where y_d is the desired output and $\hat{y}$ is the RT2FNN-A's output. Using the gradient descent algorithm, the parameters updated law is

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \Delta\mathbf{W}(k) = \mathbf{W}(k) + \eta\left(-\frac{\partial E(k)}{\partial \mathbf{W}(k)}\right) \tag{22}$$

in which η is the learning rate ($0 < \eta \le 1$). $\mathbf{W} = [\mathbf{W}_w \quad \underline{\mathbf{W}} \quad \underline{\mathbf{W}}^F \quad \overline{\mathbf{W}} \quad \overline{\mathbf{W}}^F \quad \mathbf{W}_a \quad \underline{r} \quad \underline{r}^F]$ are the adjustable parameters, where $\mathbf{W}_w$ is consequent weights, $\underline{\mathbf{W}}$ and $\underline{\mathbf{W}}^F$ are parameters of lower MFs, $\overline{\mathbf{W}}$ and $\overline{\mathbf{W}}^F$ are upper MFs parameters, $\mathbf{W}_a$ is parameter in feedback layer, and $\underline{r}$ and $\underline{r}^F$ are the column vectors, i.e.,

$$\mathbf{W}_w = \left[\underline{w} \quad \overline{w}\right]^T \tag{23}$$

$$\mathbf{W}_a = \left[\underline{a} \quad \overline{a}\right]^T \tag{24}$$

$$\underline{\mathbf{W}} = \left[\underline{m}^{(l)} \quad \underline{m}^{(r)} \quad \underline{\sigma}^{(l)} \quad \underline{\sigma}^{(r)}\right]^T \tag{25}$$

$$\overline{\mathbf{W}} = \left[\overline{m}^{(l)} \quad \overline{m}^{(r)} \quad \overline{\sigma}^{(l)} \quad \overline{\sigma}^{(r)}\right]^T \tag{26}$$

$$\underline{\mathbf{W}}^F = \left[\underline{m}^{F(l)} \quad \underline{m}^{F(r)} \quad \underline{\sigma}^{F(l)} \quad \underline{\sigma}^{F(r)}\right]^T \tag{27}$$

$$\overline{\mathbf{W}}^F = \left[\overline{m}^{F(l)} \quad \overline{m}^{F(r)} \quad \overline{\sigma}^{F(l)} \quad \overline{\sigma}^{F(r)}\right]^T . \tag{28}$$

Considering $\partial E(k) / \partial \mathbf{W}(k)$, we have

$$\frac{\partial E(k)}{\partial \mathbf{W}(k)} = \frac{\partial E(k)}{\partial \hat{y}(k)} \frac{\partial \hat{y}(k)}{\partial \mathbf{W}(k)} = -\left[y_d(k) - \hat{y}(k)\right] \frac{\partial \hat{y}(k)}{\partial \mathbf{W}(k)}. \tag{29}$$

Thus, (21) can be rewritten as

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \eta e(k) \frac{\partial \hat{y}(k)}{\partial \mathbf{W}(k)} \tag{30}$$

where $e(k) = y_d(k) - \hat{y}(k)$. The remaining work involves finding the corresponding partial derivative with respect to each parameter.

Observing equation (16) and if $j \le L$, only the term of $\sum_{j=1}^{L}\left(\overline{O}_j^{(3)} \underline{w}_j\right)$ should be considered, and only consider $\sum_{j=L+1}^{M}\left(\underline{O}_j^{(3)} \underline{w}_j\right)$ if $j > L$. Moreover, we consider $\sum_{j=1}^{R}\left(\underline{O}_j^{(3)} \overline{w}_j\right)$ if $j \le R$ in (17), as well as $\sum_{j=R+1}^{M}\left(\overline{O}_j^{(3)} \overline{w}_j\right)$ where $j > R$. Thus, we should notice the values of *j*, *R*, and *L* in deriving the update laws.

In order to avoid unnecessary tuning, we must also consider the firing regions of MFs for input variable x_i. For example, considering an upper MF as shown in Fig. 3, region (I)- $x_i \le \overline{m}_{ij}^{(l)}$, only $\overline{m}_{ij}^{(l)}$ and $\overline{\sigma}_{ij}^{(l)}$ are updated; region (II)- $\overline{m}_{ij}^{(r)} \le x_i$, only $\overline{m}_{ij}^{(r)}$ and $\overline{\sigma}_{ij}^{(r)}$ must be updated as well. Finally, region (III)- $\overline{m}_{ij}^{(l)} < x_i < \overline{m}_{ij}^{(r)}$, nothing should be done. Therefore, we can tune one side of MF for each training pattern. The results of lower MFs are the same as above.

Besides, parameter $\underline{r}$ must be updated for all three regions. Owing the recurrent property, the real time recurrent learning algorithm (RTRL) is used [26]. The update rules of RT2FNN-A can be derived. Herein, we omit them due to the limitation of writing space.

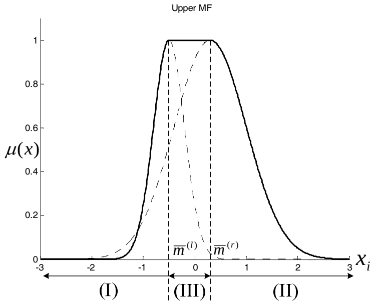


Figure 3: Firing regions definition of input variable x_i (upper MF).

B. Stability Analysis of RT2FNN-A System

By [3, 15, 23], using the Lyapunov stability approach, we have the following convergence theorem.

Theorem 1: Let $[\eta_w \quad \eta_a \quad \underline{\eta} \quad \overline{\eta} \quad \eta_{\underline{r}}]$ be the learning rates of the tuning parameters for RT2FNN-A. The asymptotic convergence of RT2FNN-A is guaranteed if proper learning rates $[\eta_w \quad \eta_a \quad \underline{\eta} \quad \overline{\eta} \quad \eta_{\underline{r}}]$ are chosen satisfying the following condition.

$$\left(\lambda_w + \underline{\lambda} + \overline{\lambda} + \lambda_{\underline{r}} + \underline{\lambda}_a + \overline{\lambda}_a + \underline{\lambda}^F + \overline{\lambda}^F + \lambda_{\underline{r}^F}\right) < 2 \qquad (31)$$

Proof: The proof is omitted due to the limitation of writing space.

IV. Nonlinear Channel Equalization Using RT2FNN-A System

To demonstrate the performance of RT2FNN-A system, several simulations regarding signal processing are constructed. The RT2FNN-A system is applied to nonlinear time-varying channel equalization to overcome the multi-path problem when signal is transmitted.

A. Nonlinear Channel Equalization

The channel characteristic represents random temporal fluctuations by the time-varying amplitude factor. The received signal can be described as follows

$$\hat{x}(t) = c_1 s(t) + c_2 s(t-1) + ... + c_p s(k-p+1) + n(k) \qquad (32)$$

where $s(t)$ is transmitted signal, and $\hat{x}(t)$ denotes the channel state; c_l, $l = 1, 2, ..., p$, are time-varying amplitude factor, and p is the channel order. The channel characteristic is similar to nonlinear time-varying channel. In order to reduce multi-path channel interference, we firstly introduce nonlinear time-varying channel equalization in next section.

The nonlinear channel equalization is a technique used to combat some imperfect phenomenon in high-speed data transmission over channels [25]. Figure 4 shows the block diagram of a communication system that is subject to inter-symbol interference (ISI) and additive white Gaussian noise (AWGN). The transmitted input symbols $s(k)$ is independent and identically distributed discrete-time random processes taking its value {-1, +1}. The signal is sent through the channel.

In real communications, the channel is too dispersive to cause interference between successive signal samples (inter-symbol interference). It will complicate reliable transmission and reception. $\hat{x}(k)$ denotes the output of the channel. The channel function can be described as [20]

$$\hat{x}(k) = f\left(s(k), s(k-1), \cdots, s(k-p+1)\right) \qquad (33)$$

where p means the channel order. Generally, f is a nonlinear function of past transmitted signals, and the channel changes slowly but significantly over time; therefore, a nonlinear channel equalizer with adaptation ability is needed. Let $\hat{\mathbf{x}}(k) \equiv [\hat{x}(k), \cdots, \hat{x}(k-n+1)]^T$, where $\hat{\mathbf{x}}(k)$ is called channel state.

At receiving terminal, the inter-symbol interference and nonlinear distortion are introduced by the channel; received signals $x(k)$ are also assumed to be corrupted by a additive noise $n(k)$, that is

$$x(k) = \hat{x}(k) + n(k) \qquad (34)$$

where $n(k)$ is an additive white Gaussian noise (AWGN), and is assumed to be zero mean white Gaussian.

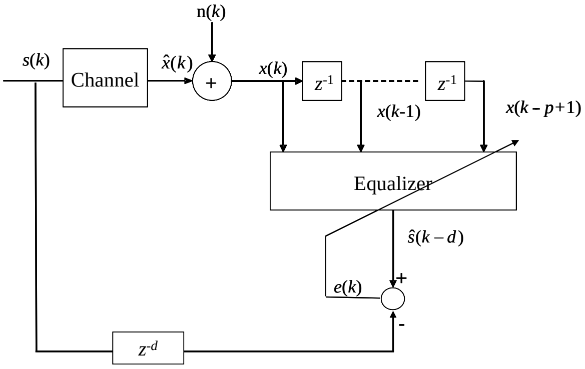


Figure 4: Block diagram of adaptive equalizer.

The function of the equalizer is to re-construct the transmitted signal, $s(k-d)$ (d denotes the decision delay), from the observed information sequence, $x(k), \cdots, x(k-p+1)$ from which greater speed and reality can be achieved. Thus, the mathematical representation of equalizer is

$$\hat{s}(k-d) = \psi(x(k), x(k-1), \cdots, x(k-p+1)) \qquad (35)$$

where $\psi : \Re^p \to \{1, -1\}$. We can say that a correct decision by the equalizer if

$$\hat{s}(k-d) = s(k-d)\,. \qquad (36)$$

Based on the category of $s(k-d)$ (i.e., ± 1), the channel states $\hat{x}(k)$ can be partitioned into two classes [2]

$$X^+ = \left\{\, \hat{x}(k) \mid s(k-d) = 1 \,\right\}, \qquad (37)$$

$$X^- = \left\{\, \hat{x}(k) \mid s(k-d) = -1 \,\right\}. \qquad (38)$$

The numbers of elements in X^+ and X^- are denoted as n_s^+ and n_s^-, respectively [2]. The probabilities for $s(k-d) = 1$ and $s(k-d) = -1$ are the same, which means, $n_s^+ = n_s^- = n_s / 2$, where n_s is the total number of channel state. Besides, the channel states in X^+ and X^- are denoted $\hat{x}_i^+ (i = 1, \cdots, n_s^+)$ and $\hat{x}_i^- (i = 1, \cdots, n_s^-)$, respectively.

Suppose channel order is $p = 2$ in the nonlinear channel function. For a time-varying channel, the coefficients of the channel, c_i, $i = 0, 1, \ldots, n$, are unknown. The nonlinear time-varying channel model is described as [18, 20]

$$x(k) = c_1 s(k) + c_2 s(k-1) - H(k) + n(k) \tag{39}$$

where c_1 and c_2 are time-varying coefficients, and $H(k)$ is Co-Channel Interference, (CCI) is described as

$$H(z) = \lambda(c_{11}(k) + c_{12}(k) z^{-1}) \tag{40}$$

where $c_{11}(k)$ and $c_{12}(k)$ are co-channel time-varying coefficients.

The time-varying coefficients, c_1 and c_2, are simulated by using a second-order Markov model. It is also called second-order Butterworth low-pass filter (LPF) which is derived by white Gaussian noise source [18]. In our simulations below, we use the function *butter*, provided by the Matlab- Signal Processing Toolbox, to generate a second-order low-pass digital Butterworth filter with cutoff frequency 0.1. Then the function filter is used to generate a colored Gaussian sequence, which is then used as time-varying channel coefficients. Note that we center $c_1(k)$ around 1 and $c_2(k)$ around 0.5 as shown in Fig. 5. The input to Butterworth filter is a white Gaussian sequence code for time-varying coefficients with length of 1000. The Matlab codes are [18]

```
[B, A] = butter(2,0.1)   % B (numerator) and A (denominator) of LPF
         c1 = 1 + filter(B , A ,  0.5*randn(1,1000))
          c2 = 0.5 + filter(B , A ,  0.25*randn(1,1000))
```

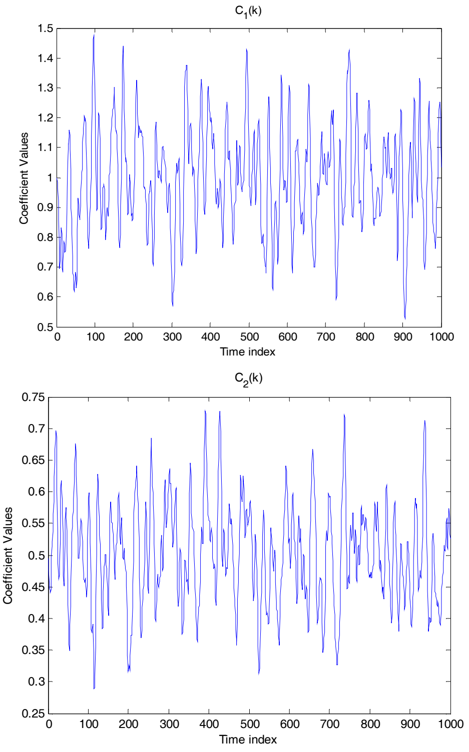


Figure 5: Time-varying channel coefficients.

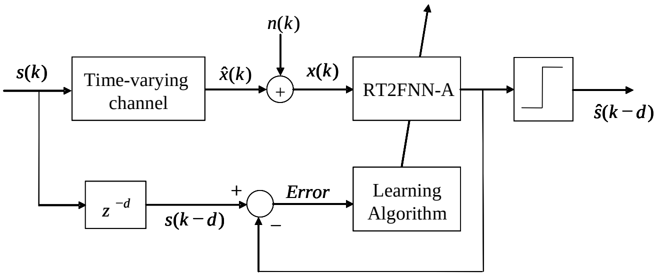


Figure 6: Block diagram of adaptive equalizer using RT2FNN-A system.

Herein, we use the RT2FNN-A system to be an adaptive equalizer for time-varying channel equalization. As Fig. 6 shows, we use the RT2FNN-A filter to construct the equalizer and take the error to update the parameters of RT2FNN-A filter which can achieve the adaptive equalizer. In our simulations, we choose the independent input sequence $s(k)$ which consists of 2000 symbols. The first 1000 symbols are used for training, and the remaining 1000 are used for testing. After training, the parameters of the T2FNN, T2FNN-A and RT2FNN-A filters are fixed, and then the testing is performed. Then, we compare two examples among these three types of filters.

Firstly, we do not consider the CCI, i.e., $H(k)$=0. We assume that the time-varying channel is the form of (39) and we choose 4 rules to construct the RT2FNN-A filter. The learning rate is chosen to be 0.1, whereas the training epoch is 50. Figure 7 shows the simulation results (solid-line: RT2FNN-A, dashed-line: T2FNN-A, and dotted-line: T2FNN). The comparisons of network structure and bit error rate (BER) are shown in Table 1. Obviously, the performance using our approach is also better than T2FNN-A and T2FNN (smaller BER value).

Next, we consider the time-varying channel with the co-channel for CCI [20]

$$H(z) = 0.9 \cdot (c_{11}(k) + c_{12}(k) z^{-1})^3 \tag{41}$$

where the nominal values are $c_{11}(k) = 1$ and $c_{12}(k) = 0.5$. The learning rate is set as 0.1, whereas the training epoch is 50. Figure 8 shows the simulation results (solid-line: RT2FNN-A, dashed-line: T2FNN-A, and dotted-line: T2FNN). We found that the performance is better than T2FNN-A and T2FNN (smaller BER value). We can see that our approach results better performance and has advantages of fewer adjustable parameters and smaller BER value.

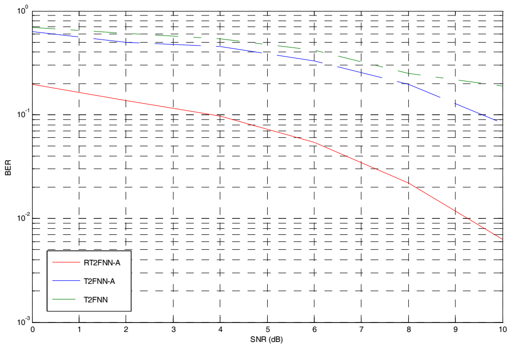


Figure 7: Performance comparisons of nonlinear time-varying channel without CCI (solid-line: RT2FNN-A, dashed-line: T2FNN-A, and dotted-line: T2FNN).

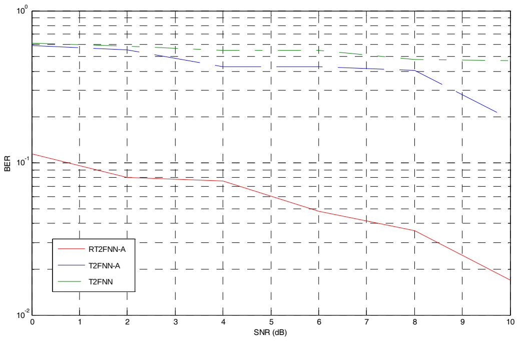


Figure 8: Performance comparisons of nonlinear time-varying channel with CCI (solid-line: RT2FNN-A, dashed-line: T2FNN-A, and dotted-line: T2FNN).

Table 1: Comparison results of network structure, rule number, parameter number, and BER for Example 1 (SNR=10 dB).

	Network structure	Rule No. (*M*)	Parameter No	BER	
				Without CCI	With CCI
T2FNN[29]	2-30-15-1	15	120	0.1897	0.4704
	2-8-4-1	4	32	0.597	0.798
T2FNN-A[37]	2-18-9-1	9	108	0.0825	0.1932
	2-8-4-1	4	48	0.434	0.601
RT2FNN-A	2-8-4-1	4	100	0.0063	0.017

V. Conclusion

In this paper, we propose a recurrent interval type-2 fuzzy neural network with asymmetric membership functions (RT2FNN-A). The novel RT2FNN-A uses the interval asymmetric type-2 fuzzy sets implements the FLS in a five-layer neural network structure which contains four-layer forward network and a feedback layer. According to the Lyapunov theorem and gradient descent method, the convergence of RT2FNN-A is guaranteed and the corresponding learning algorithm is derived. The effect of RT2FNN-A has been introduced by several illustration examples.

From the simulation results, a RT2FNN-A equalizer over various channel models are presented in this paper. Simulation results have been carried out in both real-valued and complex-valued nonlinear channels to ensure the flexibility of the proposed equalizer. The feedback layer of proposed RT2FNN-A makes it have advantages of storing past information. Moreover, the RT2FNN-A can use a small number of tuning parameters than the feed-forward fuzzy neural networks to obtain better performances (smaller BER). To reduce the computation complexity, RT2FNN-A is a good choice.

References

[1] O. Castillo and P. Melin, “Adaptive Noise Cancellation Using Type-2 Fuzzy Logic and Neural Networks,” *IEEE International Conf. on Fuzzy Systems*, Vol. 2, pp. 1093-1098, 2004.

[2] S. Chen, B. Mulgrew, and S. McLaughlin, “A Clustering Technique for Digital Communication Channel Equalization Using Radial Basis Function Network,” *IEEE Trans. on Neural Networks*, Vol. 4, No. 2, pp. 570-579, 1993.

[3] Y. C. Chen and C. C. Teng, “A Model Reference Control Structure Using A Fuzzy Neural Network,” *Fuzzy Sets and Systems*, Vol. 73, pp. 291-312, 1995.

[4] R. K. Crane, “Ionospheric Scintillation,” *Proc. of IEEE*, Vol. 65, No. 2, pp. 180-199, 1977.

[5] J. S. Jang, “ANFIS: Adaptive-network-based Fuzzy Inference System,” *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 23, No. 3, pp. 665-685, 1993.

[6] R. John and S. Coupland, “Geometric Type-1 and Type-2 Fuzzy Logic Systems,” *IEEE Trans. on Fuzzy Systems, Special Issue on Type-2 Fuzzy Systems*, Vol. 15, No. 1, pp. 3-15, 2007.

[7] C. F. Juang, ”A TSK-Type Recurrent Fuzzy Network for Dynamic Systems Processing by Neural Network and Genetic Algorithms,” *IEEE Trans. on Fuzzy Systems*, Vol. 10, No. 2. pp. 155-170, 2002.

[8] N. N. Kamik, J. Mendel, and Q. Liang, “Type-2 Fuzzy Logic Systems,” *IEEE Trans. on Fuzzy Systems*, Vol. 7, No. 6, pp. 643-658, 1999.

[9] C. H. Lee and H. Y. Pan, “Enhancing the Performance of Neural Fuzzy Systems Using Asymmetric Membership Functions,” Revised in *Fuzzy Sets and Systems*, 2007.

[10] C. H. Lee and H. Y. Pan, “Construction of Asymmetric Type-2 Fuzzy Membership Functions and Application in Time Series Prediction,” *International Conf. on Machine Learning and Cybernetics*, Vol. 4, pp. 2024-2030, 2007.

[11] C. H. Lee and Y. C. Lin, “An Adaptive Type-2 Fuzzy Neural Controller for Nonlinear Uncertain Systems,” *International Journal of Control and Intelligent*, Vol. 12, No. 1, pp. 41-50, 2005.

[12] C. H. Lee and Y. C. Lin, “Control of Nonlinear Uncertain Systems Using Type-2 Fuzzy Neural Network and Adaptive Filter,” *IEEE International Conf. on Networking, Sensing, and Control*, Vol. 2, pp. 1177-1182, 2004.

[13] C. H. Lee, Y. C. Lin, and W. Y. Lai, “Systems Identification Using Type-2 Fuzzy Neural Network (Type-2 FNN) Systems,” *IEEE International Sym. on Computational Intelligence in Robotics and Automation*, Vol. 3, pp. 1264-1269, 2003.

[14] C. H. Lee, J. L. Hong, Y. C. Lin, and W. Y. Lai, “Type-2 Fuzzy Neural Network Systems and Learning,” *International Journal of Computational Cognition*, Vol. 1, No. 4, pp. 79-90, 2003.

[15] C. H. Lee and C. C. Teng, “Identification and Control of Dynamic Systems Using Recurrent Fuzzy Neural Networks,” *IEEE Trans. on Fuzzy Systems*, Vol. 8, No. 4, pp. 349-366, 2000.

[16] A. Q. Liang and J. M. Mendel, “Interval Type-2 Fuzzy Logic Systems: Theory and Design,” *IEEE Trans. on Fuzzy Systems*, Vol. 8, No. 5, pp. 535-550, 2000.

[17] A. Q. Liang and J. M. Mendel, “Overcoming Time-varying Co-channel Interference Using Type-2 Fuzzy Adaptive Filters,” *IEEE Trans. on Circuits and Systems*, Vol. 47, No. 12, pp. 1419-1428, 2000.

[18] A. Q. Liang and J. M. Mendel, “Equalization of Nonlinear Time-varying Channels Using Type-2 Fuzzy Adaptive Filters,” *IEEE Trans. on Fuzzy Systems*, Vol. 8, No. 5, pp. 551-563, 2000.

[19] R. C. Lin, W. D. Weng and C. T. Hsueh, “Design of An SCRFNN-based Nonlinear Channel Equaliser,” *IEE Proc. Communications*, Vol. 152, No. 6, pp. 771-779, 2005.

[20] Y. C. Lin, *Type-2 Fuzzy Neural Network System and Its Applications in Signal Processing, Nonlinear System Identification, and Control*, Master Paper of Yuan-Ze University, Taiwan, 2004.

[21] J. M. Mendel and R. I. John, “Type-2 Fuzzy Sets Made Simple,” *IEEE Trans. on Fuzzy Systems*, Vol. 10, No. 2, pp. 117-127, 2002.

[22] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Upper Saddle River, Prentice-Hall, NJ, 2001.

[23] H. Y. Pan, *Type-2 Fuzzy Neural Network With Asymmetric Membership Functions Analysis and Its Applications*, Master Paper of Yuan-Ze University, Taiwan, 2006.

[24] C. H. Wang, C. S. Cheng, and T. T. Lee, “Dynamical Optimal Training for Interval Type-2 Fuzzy Neural Network (T2FNN),” *IEEE Trans. on Systems, Man, Cybernetics Part-B*, Vol. 34, No. 3, pp. 1462-1477, 2004.

[25] R. J. Williams and D. Zipser, “A Learning Algorithm for Continually Running Recurrent Neural Networks,” *Neural Computation*, Vol. 1, No. 2, pp. 270-280, 1989.

[26] L. A. Zadeh, “The Concept of A Linguistic Variable and Its Application to Approximate Reasoning,” *Information Sciences*, Vol. 8, No.3, pp. 199-249, 1975.