Stabilization of Type-2 Fuzzy Takagi-Sugeno-Kang Identifier Using Lyapunov Functions

Erdal Kayacan*, Mojtaba Ahmadieh Khanesar† and Erkan Kayacan‡
*School of Mechanical & Aerospace Engineering, Nanyang Technological University, 639798, Singapore
E-mail: erdal@ntu.edu.sg
†Faculty of Electrical and Computer Engineering, Semnan University, Semnan, Iran
E-mail: ahmadieh@semnan.ac.ir
‡Delft Center for Systems and Control, Delft University of Technology, The Netherlands
E-mail: erkan.kayacan@ieee.org

***Abstract*—Differing from previous studies, where sliding mode control theory-based rules are proposed for only the consequent part of the network, the developed algorithm in this paper applies fully sliding mode parameter update rules for both the premise and consequent parts of the interval type-2 fuzzy neural networks. The stability of the proposed learning algorithm has been proved by using an appropriate Lyapunov function. Then, the performance of the proposed learning algorithm is tested on the identification of wing flutter data set available online as a benchmark system and the prediction of Mackey-Glass chaotic system. The simulation results indicate that the proposed algorithm is significantly faster than the gradient-based methods as well as providing a slightly better identification performance. The reason for the fast convergence is that the proposed parameter update rules do not have any matrix manipulations which makes them simple to be implemented in real-time systems. In addition, the responsible parameter for sharing the contributions of the lower and upper parts of the type-2 fuzzy membership functions is also tuned. Another prominent feature of the proposed learning algorithm is to have a closed form which makes it easier to implement than the other existing learning methods, e.g. gradient-based methods.**

***Index Terms*—Type-2 fuzzy neural networks, sliding mode learning algorithm, nonlinear system identification.**

I. Introduction

Type-2 fuzzy logic systems (T2FLSs) are proposed as the extensions of type-1 fuzzy logic systems (T1FLSs) in order to be able to model uncertainties that invariably exist in the rule base of the system and deal with noise [1]–[4]. Whereas membership functions (MFs) are totally certain in type-1 fuzzy sets, they are themselves fuzzy in type-2 fuzzy sets. The latter case results in a fact that the antecedent and consequent parts of the rules are uncertain. As there are infinite type-1 fuzzy MFs in the footprint of uncertainty of a type-2 fuzzy MF, it is believed that the T2FLSs have the ability of modeling uncertainties in the rule base better than their type-1 counterparts. Therefore, T2FLSs appear to be a more promising method for handling uncertainties such as noisy data and variable working conditions both in modeling and control purposes [5].

Fuzzy neural networks (FNNs) are shown to be more powerful tools that combine the capability of fuzzy reasoning to handle uncertain information and the capability of artificial neural networks (ANNs) to learn from input-output data sets in modeling nonlinear dynamic systems. There are number of algorithms for the training of FNNs in literature. For instance, whereas the gradient descent (GD) algorithm is a well-known optimization method to tune the parameters of both ANNs and FNNs [6], there are some drawbacks in this algorithm. The most important issue is that the selection of smaller learning rate is critical in GD algorithm as the learning can get stuck in a local minima. Moreover, some instability issues may occur if the value of learning rate is chosen too large. Especially, in FNNs, another issue for a GD algorithm is that the obtaining of the parameter update rules for the premise part of the network is very complex. As an alternative to GD-based methods, the use of evolutionary approaches has also been suggested [7]. However, as these methods have some stochastic operators such as mutation, crossover, etc., it is almost impossible to show the stability of learning. Furthermore, there is no analytical way to choose the optimal parameters of the aforementioned stochastic operators.

In order to cope with the mentioned issues above, sliding mode control (SMC) theory-based algorithms are derived to tune the parameters of ANNs and type-1 FNNs (T1FNNs) [8], [9]. As SMC theory-based learning algorithms do not need any partial derivatives and matrix manipulations in the adaptation laws [10], [11], they are computationally more efficient than the traditional learning techniques in online tuning of ANNs and FNNs [12], [13]. Thus, the parameter update rules are much simpler when compared to other algorithms, such as GD-based methods. Furthermore, since SMC theory-based learning algorithms benefit from mathematical stability analysis, they are more robust against the parameter uncertainties in the system. Motivated by the successful results of these learning algorithms in T1FNNs, similar derivations for the training of type-2 FNNs (T2FNNs) are also proposed [14], [15].

In this paper, the major contributions to the T2FNNs are as follows: the first is the proposal of fully SMC theory-based learning rules whereas all similar studies in literature consider SMC theory-based rules for only limited number of parameters. For instance, the parameter update rules for the center values of the type-2 fuzzy MFs in [14] do not have

SMC theory-based rules. The second contribution of this paper is that the proposed algorithm tunes the sharing of the lower and upper MFs in a T2FNN which allows us to manage non-uniform uncertainties in the rule base of T2FLSs.

II. Type-2 Fuzzy Logic Systems (T2FLSs)

A. T2FLSs Overview

A first-order interval type-2 Takagi-Sugeno-Kang (TSK) fuzzy *if-then* rule base with I input variables is preferred in this investigation. Where the consequent parts are crisp numbers, the premise parts are type-2 fuzzy functions. The r^{th} rule is as follows:

R_r : If x_1 is $\widetilde{A}_{1j}$ … and x_i is $\widetilde{A}_{ik}$ and… and x_I is $\widetilde{A}_{II}$ then

$$f_r = \sum_{i=1}^{I} a_{ri}x_i + b_r \tag{1}$$

where $x_i(i = 1...I)$ are the inputs of the type-2 TSK model, $\widetilde{A}_{ik}$ is the k^{th} type-2 fuzzy MF $(k = 1...K)$ corresponding to the input i^{th} variable, K is the number of MFs for the i^{th} input which can be different for each input. The parameters a_r and b_r stand for the consequent part and $f_r(r = 1...N)$ is the output function.

The upper and lower type-2 fuzzy Gaussian MFs with an uncertain standard deviation (Fig. 1) can be represented as follows:

$$\overline{\mu}_{ik}(x_i) = exp\left(-\frac{1}{2}\frac{(x_i - c_{ik})^2}{\overline{\sigma}_{ik}^2}\right) \tag{2}$$

$$\underline{\mu}_{ik}(x_i) = exp\left(-\frac{1}{2}\frac{(x_i - c_{ik})^2}{\underline{\sigma}_{ik}^2}\right) \tag{3}$$

where c_{ik} is the center value of the k^{th} type-2 fuzzy set for the i^{th} input. The parameters $\overline{\sigma}_{ik}$ and $\underline{\sigma}_{ik}$ are standard deviations for the upper and lower MFs.

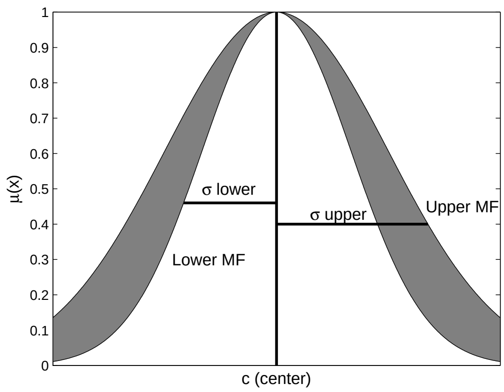

Fig. 1: Type-2 Gaussian fuzzy MF with uncertain standard deviation

B. Interval Type-2 A2-CO TSK Model

The structure used in this investigation is called A2-C0 fuzzy system [16]. In such a structure, first, the lower and upper membership degrees $\underline{\mu}$ and $\overline{\mu}$ are determined for each input signal being fed to the system. Next, the firing strengths of the rules using the *prod* t-norm operator are calculated as follows:

$$\underline{w}_r = \underline{\mu}_{\tilde{A}1}(x_1) * \underline{\mu}_{\tilde{A}2}(x_2) * \cdots \underline{\mu}_{\tilde{A}I}(x_I)$$
$$\overline{w}_r = \overline{\mu}_{\tilde{A}1}(x_1) * \overline{\mu}_{\tilde{A}2}(x_2) * \cdots \overline{\mu}_{\tilde{A}I}(x_I) \tag{4}$$

The consequent part corresponding to each fuzzy rule is a linear combination of the inputs $x_1, x_2 ... x_I$. This linear function is called f_r and is defined as in (1). The output of the network is calculated as follows:

$$y_N = q\sum_{r=1}^{N} f_r\underline{\widetilde{w}}_r + (1-q)\sum_{r=1}^{N} f_r\widetilde{\overline{w}}_r \tag{5}$$

where $\widetilde{\underline{w_r}}$ and $\widetilde{\overline{w_r}}$ are the normalized values of the lower and the upper output signals from the second hidden layer of the network as follows:

$$\widetilde{\underline{w_r}} = \frac{\underline{w_r}}{\sum_{i=1}^{N}\underline{w_r}} \text{ and } \widetilde{\overline{w_r}} = \frac{\overline{w_r}}{\sum_{i=1}^{N}\overline{w_r}} \tag{6}$$

The design parameter, q, weights the sharing of the lower and upper firing levels of each fired rule. This parameter can be a constant (equal to 0.5 in most cases) or a time varying parameter. In this investigation, the latter is preferred. In other words, the parameter update rules and the proof of the stability of the learning process are given for the case of an adaptable q.

The following vectors can be specified:
$\underline{\widetilde{W}}(t) = \left[\widetilde{\underline{w_1}}(t)\ \widetilde{\underline{w_2}}(t) ... \widetilde{\underline{w_N}}(t)\right]^T$,
$\widetilde{\overline{W}}(t) = \left[\widetilde{\overline{w_1}}(t)\ \widetilde{\overline{w_2}}(t) ... \widetilde{\overline{w_N}}(t)\right]^T$ and $F = [f_1\ f_2 ... f_N]$

The following assumptions have been made in this investigation: The time derivative of both the input signals and output signal can be considered bounded:

$$|\dot{x}_i(t)| \leq B_{\dot{x}},\ min(x_i^2(t)) = B_{x^2},\ (i = 1 \ldots I) \text{ and } |\dot{y}(t)| \leq B_{\dot{y}}\ \forall t \tag{7}$$

where $B_{\dot{x}}$, B_{x^2} and $B_{\dot{y}}$ are assumed to be some known positive constants.

III. Sliding Mode Control Theory-based Learning Algorithm

The zero value of the learning error coordinate can be defined as a time-varying sliding surface in (8). The condition defined in (8) guarantees that when the system is on the sliding surface, the output of the network, $y_N(t)$, will perfectly follow the desired output signal, $y(t)$, for all time $t > t_h$. The time instant t_h is defined to be the reaching time for being $e(t) = 0$.

$$S\big(e(t)\big) = e(t) = y_N(t) - y(t) = 0 \tag{8}$$

Definition: A sliding motion will appear on the sliding manifold $S(e(t)) = e(t) = 0$ after a time t_h, if the condition $S(t)\dot{S}(t) < 0$ is satisfied for all t in some nontrivial semi-open subinterval of time of the form $[t, t_h) \subset (0, t_h)$.

The parameter update rules for the T2FNN proposed in this paper are given by the following theorem.

Theorem 1: If the adaptation laws for the parameters of the considered T2FNN are chosen as:

$$\dot{c}_{ik} = \dot{x}_i + (x_i - c_{ik})\alpha_1 \mathrm{sgn}(e) \tag{9}$$

$$\dot{\underline{\sigma}}_{ik} = -\left(\underline{\sigma}_{ik} + \frac{(\underline{\sigma}_{ik})^3}{(x_i - c_{ik})^2}\right)\alpha_1 \mathrm{sgn}(e) \tag{10}$$

$$\dot{\overline{\sigma}}_{ik} = -\left(\overline{\sigma}_{ik} + \frac{(\overline{\sigma}_{ik})^3}{(x_i - c_{ik})^2}\right)\alpha_1 \mathrm{sgn}(e) \tag{11}$$

$$\dot{a}_{ri} = -x_i \frac{q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r}{(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)^T (q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)} \alpha \mathrm{sgn}(e) \tag{12}$$

$$\dot{b}_r = -\frac{q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r}{(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)^T (q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)} \alpha \mathrm{sgn}(e) \tag{13}$$

$$\dot{q} = -\frac{1}{F(\underline{\widetilde{W}} - \widetilde{\overline{W}})^T} \alpha sgn(e) \tag{14}$$

where α is taken as follows:

$$\alpha \geq \frac{(IB_a B_{\dot{x}} + B_{\dot{y}})}{2 + IB_{x^2}} \tag{15}$$

then, given an arbitrary initial condition $e(0)$, the learning error $e(t)$ will converge to zero within a finite time t_h.

Proof: The reader is referred to Appendix.

Remark: Since the output of the T2FNN is quite sensitive to the changes in the parameters of the antecedent parts, different values for the learning rates of the antecedent and consequent part parameters are used. In other words, a smaller value (α_1) is chosen for the antecedent parts.

IV. Simulation Studies

A system identification process is the finding of a mathematical relationship between the input and the output of the system. In this study, time delayed inputs and outputs of the system are fed into the identifier. The SMC theory-based learning algorithm is used to estimate the parameters of the T2FNN structure that the difference between the plant output $y(k)$ and the identifier output $y_N(k)$ will be minimum for all input values of $u(k)$.

As a performance criterion in the simulation studies, the root-mean-squared-error (RMSE) given in (16) is used:

$$RMSE = \sqrt{\frac{\sum_{k=1}^{K}(y(k) - y_N(k))^2}{K}} \tag{16}$$

where K is the number of samples.

It is to be noted that even if the network structure is the same with the one in [17], the proposed learning rules in this investigation are completely novel and fully sliding mode. In all the examples in this section, the network is designed with three inputs and one output. The inputs of the identifier are the input signal to the plant and the two delayed signals from the plant output with a discretization period T_o of 1ms. Each input is fuzzified by using three Gaussian type-2 fuzzy MFs with a fixed center and uncertain standard deviation. To be able to make a fair comparison, each experiment has been realized for ten times with a random initialization of the network parameters, and the average numbers are given in the paper.

A. Example 1: Wing flutter data set

To compare the identification performance of both the proposed learning algorithm and the conventional GD method, an existing benchmark data set is used: wing flutter data set available online [18]. Whereas Fig. 3(a) demonstrates the output of the model and the real-time system for the training data only, Fig. 3(b) shows the response of the system for the test data. RMSE values versus epoch number which indicates a stable learning with the proposed learning algorithm is also presented in Fig. 3(c). As can been from these figures, the T2FNN gives accurate modeling results. Thanks to the novel fully sliding mode parameter update rules in this paper, the presented results are significantly better when compared to the ones in [19]. In Fig. 3(d), the adaptation of the parameter q is presented which is also learnt by the proposed algorithm. By doing so, the contributions of the upper and lower MFs are also tuned during the simulations.

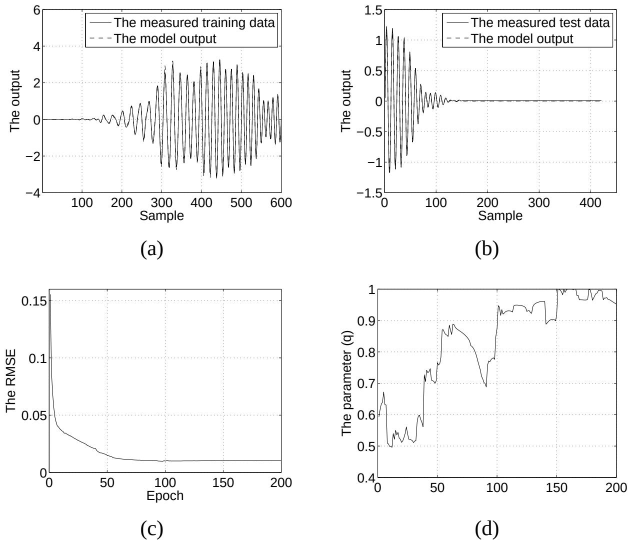

Fig. 2: The output of the model and the T2FNN for the training data (a) The output of the model and the T2FNN for the test data (b) RMSE versus epoch number (c) The adaptation of the parameter q (d) for the wing flutter data set

B. Example 2: Prediction of chaotic Mackey-Glass time series

As another comparison for the identification performance of both the proposed learning algorithm and conventional GD method, one more existing benchmark system is used: Mackey-Glass chaotic system. Both algorithms are used to predict the noisy chaotic Mackey-Glass time series. This chaotic system is a well-known benchmark problem in the literature described by the following dynamic equation [20]:

$$\dot{x}(t) = 0.2\frac{x(t-\tau)}{1 + x^{10}(t-\tau)} - 0.1x(t) \tag{17}$$

The numerical values selected for the chaotic system above are $\tau = 17$, $x(0) = 1.2$ in this study.

The predictor goal is to predict $x(t+1)$ using the inputs $x(t-3), x(t-2), x(t-1)$ and $x(t)$. For each input, two MFs are used. The number of training data is selected as 600, and the number of test data is 476. As the measure of noise level, signal to noise ratio (SNR) is used.

Figure 3 shows the convergence graphs of the proposed learning algorithm for Mackey-Glass time series in the presence of white noise (SNR=50dB). As can be seen from these figures, the convergence performance of the proposed method is quite satisfactory even under noisy conditions.

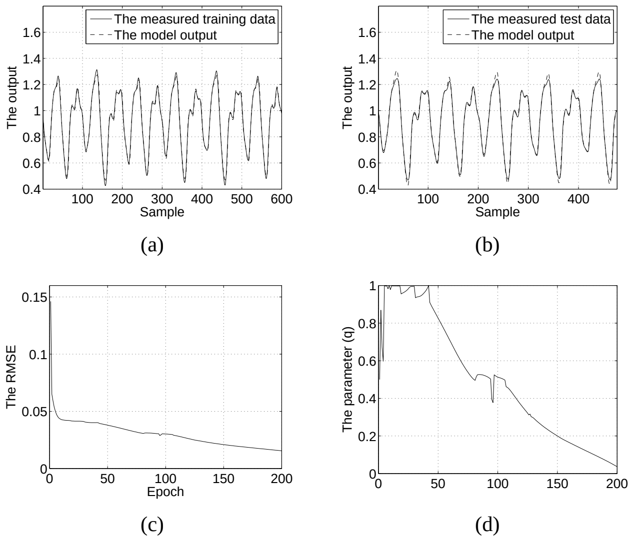

Fig. 3: The output of the model and the T2FNN for the training data(a) The output of the model and the T2FNN for the test data (b) RMSE versus epoch number (c) The adaptation of the parameter q (d) for the Mackey-Glass system

V. Analysis and Discussion

In Tables I and II, the comparison of GD technique and the proposed SMC theory-based learning method is given with respect to their identification performance and computation time. It can be seen that the identification performance of the proposed learning algorithm is slightly better than GD. Moreover, the computation time of the proposed SMC theory-based learning algorithm is significantly lower than the GD method. This conclusion results in a fact that the proposed method in this paper is more practical in real-time applications.

GD method includes the calculations of the partial derivatives of the output with respect to the parameters which is a very difficult task, and do not have any closed (explicit) form. This feature brings a lot of difficulties during the coding of the GD method. On the other hand, as can be seen from the parameter update rules proposed in this paper, the algorithm has a closed form. This results in a simpler and easier to debug the coding process of the proposed identifier.

TABLE I: Comparison of different learning techniques for wing flutter data set

	Performance		
	Training	Testing	Computation time (s)
SMC-based learning	0.0135	0.0036	56.03
GD	0.0361	0.0112	77.10

TABLE II: Comparison of different learning techniques for chaotic Mackey-Glass system

	Performance		
	Training	Testing	Computation time (s)
SMC-based learning	0.0207	0.0225	47.80
GD	0.0297	0.1745	67.48

VI. Conclusions

A novel fully sliding mode parameter update rules have been proposed for the training of interval T2FNNs for the identification of a real-time wing flutter data set and the prediction of chaotic Mackey-Glass time series. The simulation results are promising for the potential of the proposed structure in real time systems since the computation time of the proposed algorithm is significantly lower than the GD method. The reason for the fast convergence is that the proposed parameter update rules do not have any matrix manipulations which makes them simple to be implemented in real-time systems. It is to be noted that these parameter update rules can also be used for the control purposes in which the computation time is prominent.

Appendix A
Proof of Theorem 1

The time derivative of (6) is calculated as follows:

$$\dot{\underline{\tilde{w}}}_r = -\underline{\tilde{w}}_r \underline{K}_r + \underline{\tilde{w}}_r \sum_{r=1}^{N} \underline{\tilde{w}}_r \underline{K}_r; \; \dot{\overline{\tilde{w}}}_r = -\overline{\tilde{w}}_r \overline{K}_r + \overline{\tilde{w}}_r \sum_{r=1}^{N} \overline{\tilde{w}}_r \overline{K}_r \tag{18}$$

where

$$\underline{A}_{ik} = \frac{x_i - c_{ik}}{\underline{\sigma}_{ik}} \text{ and } \overline{A}_{ik} = \frac{x_i - c_{ik}}{\overline{\sigma}_{ik}}$$

$$\underline{K}_r = \sum_{i=1}^{I} \underline{A}_{ik} \dot{\underline{A}}_{ik} \text{ and } \overline{K}_r = \sum_{i=1}^{I} \overline{A}_{ik} \dot{\overline{A}}_{ik}$$

If (9)-(11) are inserted into the equations above, (19) can be obtained:

$$\overline{K}_r = \underline{K}_r = \sum_{i=1}^{I} \overline{A}_{ik} \dot{\overline{A}}_{ik} = \sum_{i=1}^{I} \underline{A}_{ik} \dot{\underline{A}}_{ik} = I\alpha \operatorname{sgn}(e) \tag{19}$$

By using the following Lyapunov function, the stability condition is checked as follows:

$$V = \frac{1}{2} e^2 \tag{20}$$

The time derivative of (20) can be calculated as follows:

$$\dot{V} = \dot{e}e = e(\dot{y}_N - \dot{y}) \quad (21)$$

Differentiating (5), the following term can be obtained:

$$\begin{aligned}\dot{y}_N &= \dot{q}\sum_{r=1}^{N} f_r \underline{\tilde{w}}_r + q\sum_{r=1}^{N}(\dot{f}_r \underline{\tilde{w}}_r + f_r \dot{\underline{\tilde{w}}}_r) - \dot{q}\sum_{r=1}^{N} f_r \tilde{\overline{w}}_r \\ &+ (1-q)\sum_{r=1}^{N}(\dot{f}_r \tilde{\overline{w}}_r + f_r \dot{\tilde{\overline{w}}}_r) \quad (22)\end{aligned}$$

By using (18), (19) and (22), the following term can be obtained:

$$\begin{aligned}\dot{y}_N &= \dot{q}\sum_{r=1}^{N} f_r \underline{\tilde{w}}_r + q\sum_{r=1}^{N}\Big(\dot{f}_r \underline{\tilde{w}}_r + f_r(-\underline{\tilde{w}}_r \underline{K}_r + \underline{\tilde{w}}_r \sum_{r=1}^{N} \underline{\tilde{w}}_r \underline{K}_r)\Big) \\ &- \dot{q}\sum_{r=1}^{N} f_r \tilde{\overline{w}}_r \\ &+ (1-q)\sum_{r=1}^{N}\Big(\dot{f}_r \tilde{\overline{w}}_r + f_r(-\tilde{\overline{w}}_r \overline{K}_r + \tilde{\overline{w}}_r \sum_{r=1}^{N} \tilde{\overline{w}}_r \overline{K}_r)\Big) \\ &= \dot{q}\sum_{r=1}^{N} f_r \underline{\tilde{w}}_r + q\sum_{r=1}^{N}\Big(\dot{f}_r \underline{\tilde{w}}_r - I\alpha \operatorname{sgn}(e)\, f_r(\underline{\tilde{w}}_r - \underline{\tilde{w}}_r \sum_{r=1}^{N} \underline{\tilde{w}}_r)\Big) \\ &- \dot{q}\sum_{r=1}^{N} f_r \tilde{\overline{w}}_r \\ &+ (1-q)\sum_{r=1}^{N}\Big(\dot{f}_r \tilde{\overline{w}}_r - I\alpha \operatorname{sgn}(e)\, f_r(\tilde{\overline{w}}_r - \tilde{\overline{w}}_r \sum_{r=1}^{N} \tilde{\overline{w}}_r)\Big) \quad (23)\end{aligned}$$

The following equation is correct by definition:

$$\sum_{r=1}^{N} \underline{\tilde{w}}_r = 1 \text{ and } \sum_{r=1}^{N} \tilde{\overline{w}}_r = 1 \quad (24)$$

By using (12), (13), (14) and (24), the following term can be achieved:

$$\begin{aligned}\dot{y}_N &= -\frac{1}{F(\underline{\tilde{W}} - \tilde{\overline{W}})^T}\alpha \operatorname{sgn}(e)\sum_{r=1}^{N} f_r(\underline{\tilde{w}}_r - \tilde{\overline{w}}_r) \\ &+ \sum_{r=1}^{N} \dot{f}_r(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r) \\ &= -\alpha \operatorname{sgn}(e) \\ &+ \sum_{r=1}^{N}\Big[\Big(\sum_{i=1}^{I}(\dot{a}_{ri}x_i + a_{ri}\dot{x}_i) + \dot{b}_r\Big) \\ &(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\Big] \quad (25)\end{aligned}$$

If (25) is inserted into the candidate Lyapunov function, (26) can be obtained:

$$\begin{aligned}\dot{V} &= \dot{e}e = e(\dot{y}_N - \dot{y}) \\ &= e\Big[-\alpha \operatorname{sgn}(e) + \sum_{r=1}^{N}\Big[\Big(\sum_{i=1}^{I}(\dot{a}_{ri}x_i + a_{ri}\dot{x}_i) + \dot{b}_r\Big) \\ &(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\Big] - \dot{y}\Big]\end{aligned}$$

$$\begin{aligned}&= e\Big[-\alpha \operatorname{sgn}(e) + \sum_{r=1}^{N}\Big[\Big(\sum_{i=1}^{I}\big(-(x_i \alpha \operatorname{sgn}(e) \\ &\frac{(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)}{(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)^T(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)})x_i + a_{ri}\dot{x}_i\big) \\ &-\alpha \operatorname{sgn}(e)\frac{(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)}{(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)^T(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)}\Big) \\ &(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\Big] - \dot{y}\Big]\end{aligned}$$

$$\begin{aligned}&= e\Big[-\alpha \operatorname{sgn}(e) \\ &+ \sum_{r=1}^{N}\Big[\sum_{i=1}^{I}\big(-\alpha \operatorname{sgn}(e)x_i^2 + a_{ri}\dot{x}_i(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\big) \\ &-\alpha \operatorname{sgn}(e)\Big] - \dot{y}\Big]\end{aligned}$$

$$\begin{aligned}&= e\Big[-2\alpha \operatorname{sgn}(e) + \sum_{r=1}^{N}\Big[\sum_{i=1}^{I}\big(-\alpha \operatorname{sgn}(e)x_i^2 \\ &+ a_{ri}\dot{x}_i(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\big)\Big] - \dot{y}\Big]\end{aligned}$$

$$\begin{aligned}&= -\mid e\mid 2\alpha + e\Big[\sum_{r=1}^{N}\Big[\sum_{i=1}^{I}\big(-\alpha \operatorname{sgn}(e)x_i^2 \\ &+ a_{ri}\dot{x}_i(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\big)\Big] - \dot{y}\Big] \quad (26)\end{aligned}$$

$$\begin{aligned}\dot{V} &< -\mid e\mid 2\alpha + \mid e\mid\Big[\sum_{r=1}^{N}\Big[\sum_{i=1}^{I}\big(-\alpha B_{x^2} \\ &+ B_a B_{\dot{x}}(q\underline{\tilde{w}}_r + (1-q)\tilde{\overline{w}}_r)\big)\Big] + B_{\dot{y}}\Big]\end{aligned}$$

$$\begin{aligned}&< -\mid e\mid 2\alpha + \mid e\mid\Big[\sum_{r=1}^{N}\Big[-I\alpha B_{x^2} + IB_a B_{\dot{x}}(q\underline{\tilde{w}}_r \\ &+ (1-q)\tilde{\overline{w}}_r)\Big] + B_{\dot{y}}\Big] \\ &< -\mid e\mid 2\alpha + \mid e\mid\Big[-I\alpha B_{x^2} + IB_a B_{\dot{x}}(q\sum_{r=1}^{N}\underline{\tilde{w}}_r \\ &+ (1-q)\sum_{r=1}^{N}\tilde{\overline{w}}_r) + B_{\dot{y}}\Big]\end{aligned}$$

$$< -\mid e\mid 2\alpha + \mid e\mid\Big[-I\alpha B_{x^2} + IB_a B_{\dot{x}} + B_{\dot{y}}\Big]$$

$$< -\mid e\mid\Big(2\alpha + I\alpha B_{x^2}\Big) + \mid e\mid\Big[IB_a B_{\dot{x}} + B_{\dot{y}}\Big] < 0 \quad (27)$$

REFERENCES

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

[2] M. Fadali, S. Jafarzadeh, and A. Nafeh, "Fuzzy tsk approximation using type-2 fuzzy logic systems and its application to modeling a photovoltaic array," in *American Control Conference (ACC), 2010*, June 2010, pp. 6454–6459.

[3] M. Khanesar, E. Kayacan, M. Teshnehlab, and O. Kaynak, "Analysis of the noise reduction property of type-2 fuzzy logic systems using a novel type-2 membership function," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 41, no. 5, pp. 1395–1406, Oct 2011.

[4] M. Khanesar, M. Teshnehlab, E. Kayacan, and O. Kaynak, "A novel type-2 fuzzy membership function: application to the prediction of noisy data," in *Computational Intelligence for Measurement Systems and Applications (CIMSA), 2010 IEEE International Conference on*, 2010, pp. 128–133.

[5] C.-F. Juang and C.-H. Hsu, "Reinforcement ant optimized fuzzy controller for mobile-robot wall-following control," *Industrial Electronics, IEEE Transactions on*, vol. 56, no. 10, pp. 3931 –3940, oct. 2009.

[6] F. Gaxiola, P. Melin, F. Valdez, and O. Castillo, "Interval type-2 fuzzy weight adjustment for backpropagation neural networks with application in time series prediction," *Information Sciences*, vol. 260, no. 0, pp. 1 – 14, 2014.

[7] D. Bernardo, H. Hagras, and E. Tsang, "A genetic type-2 fuzzy logic based system for the generation of summarised linguistic predictive models for financial applications," *Soft Computing*, vol. 17, no. 12, pp. 2185–2201, 2013.

[8] G. Parma, B. Menezes, and A. Braga, "Sliding mode algorithm for training multilayer artificial neural networks," *Electronics Letters*, vol. 34, no. 1, pp. 97 –98, 1998.

[9] Y. Shuanghe, Y. Xinghuo, and M. Zhihong, "Fuzzy sets and systems," *A fuzzy neural network approximator with fast terminal sliding mode and its applications*, vol. 148, no. 3, pp. 469–486, 11 2004.

[10] M. Khanesar, E. Kayacan, M. Teshnehlab, and O. Kaynak, "Levenberg marquardt algorithm for the training of type-2 fuzzy neuro systems with a novel type-2 fuzzy membership function," in *Advances in Type-2 Fuzzy Logic Systems (T2FUZZ), 2011 IEEE Symposium on*, April 2011, pp. 88–93.

[11] ——, "Extended kalman filter based learning algorithm for type-2 fuzzy logic systems and its experimental evaluation," *Industrial Electronics, IEEE Transactions on*, vol. 59, no. 11, pp. 4443–4455, Nov 2012.

[12] G. Cascella, F. Cupertino, A. Topalov, O. Kaynak, and V. Giordano, "Adaptive control of electric drives using sliding-mode learning neural networks," in *Industrial Electronics, 2005. ISIE 2005. Proceedings of the IEEE International Symposium on*, vol. 1, 2005, pp. 125 – 130.

[13] M. O. Efe, O. Kaynak, and X. Yu, "Sliding mode control of a three degrees of freedom anthropoid robot by driving the controller parameters to an equivalent regime," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 122, no. 4, pp. 632 –640, December 2000.

[14] E. Kayacan, O. Cigdem, and O. Kaynak, "Sliding mode control approach for online learning as applied to type-2 fuzzy neural networks and its experimental evaluation," *Industrial Electronics, IEEE Transactions on*, vol. 59, no. 9, pp. 3510–3520, 2012.

[15] E. Kayacan and O. Kaynak, "Sliding mode control theory-based algorithm for online learning in type-2 fuzzy neural networks: application to velocity control of an electro hydraulic servo system," *International Journal of Adaptive Control and Signal Processing*, vol. 26, no. 7, pp. 645–659, 2012.

[16] M. Begian, W. Melek, and J. Mendel, "Parametric design of stable type-2 tsk fuzzy systems," in *Fuzzy Information Processing Society, 2008. NAFIPS 2008. Annual Meeting of the North American*, May 2008, pp. 1–6.

[17] S. Ahmed, N. Shakev, A. Topalov, K. Shiev, and O. Kaynak, "Sliding mode incremental learning algorithm for interval type-2 Takagi - Sugeno - Kang fuzzy neural networks," *Evolving Systems*, vol. 3, no. 3, pp. 179 –188, September 2012.

[18] B. D. Moor, "Daisy: Database for the Identification of Systems," April 2014, Used dataset: Wing flutter, code 96-008. [Online]. Available: URL:http://homes.esat.kuleuven.be/ smc/daisy/

[19] D. Baptista and F. Morgado-Dias, "Comparing different implementations for the levenberg-marquardt algorithm," in *10th Portuguese Conference on Automatic Control*, Funchal, Portugal, 16-18 July 2012, pp. 15–20.

[20] M. Mackey and L. Glass, "Oscillation and chaos in physical control system," *Science*, vol. 197, pp. 287–289, 1977.