machines

Article

State Estimation of Memristor Neural Networks with Model Uncertainties

Libin Ma and Mao Wang *

Space Control and Inertial Technology Research Center, Harbin Institute of Technology, Harbin 150001, China
* Correspondence: maowang@hit.edu.cn

Abstract: This paper is concerned with the problem of state estimation of memristor neural networks with model uncertainties. Considering the model uncertainties are composed of time-varying delays, floating parameters and unknown functions, an improved method based on long short term memory neural networks (LSTMs) is used to deal with the model uncertainties. It is proved that the improved LSTMs can approximate any nonlinear model with any error. On this basis, adaptive updating laws of the weights of improved LSTMs are proposed by using Lyapunov method. Furthermore, for the problem of state estimation of memristor neural networks, a new full-order state observer is proposed to achieve the reconstruction of states based on the measurement output of the system. The error of state estimation is proved to be asymptotically stable by using Lyapunov method and linear matrix inequalities. Finally, two numerical examples are given, and simulation results demonstrate the effectiveness of the scheme, especially when the memristor neural networks with model uncertainties.

Keywords: memristor neural networks; model uncertainties; long short term memory neural networks (LSTMs); adaptive updating laws; full-order state observer

check for updates

Citation: Ma, L.; Wang, M. State Estimation of Memristor Neural Networks with Model Uncertainties. *Machines* **2022**, *10*, 1228. https://doi.org/10.3390/machines10121228

Academic Editor: Dan Zhang

Received: 24 October 2022
Accepted: 12 December 2022
Published: 15 December 2022

1. Introduction

Since the prototype of memristor was born in 1971 by Chua [1], memristor have been widely used in all walks of life. The vector–matrix multiplication is realized by the crossbar array structure of memristor, and a neural network can be realized by the corresponding coding scheme based on it. Various neural networks based on memristor hardware have been developed rapidly. Because of an incomparable advantage that memristor neural networks can reflect the memorized information, the memristor neural networks are particularly suitable for self-adaptability, nonlinear systems, self-learning, and associative storage, so memristor neural networks are widely used in brain simulation, pattern recognition, neural morphologic computation, knowledge acquisition, and various hardware applications involving neural networks [2–15]. To list a few, the experimental implementation of transistor-free metal-oxide memristor crossbars, with device variability sufficiently low to allow operation of integrated neural networks, in a simple network: a single-layer perceptron (an algorithm for linear classification) was shown in [16]. In [17], a structure suppressing the overshoot current was investigated to approach the conditions required as an ideal synapse of a neuromorphic system. In [18], fully memristive artificial neural networks were built by using diffusive memristors based on silver nanoparticles in a dielectric film. The electrical properties and conduction mechanism of the fabricated IGZO-based memristor device in a 10 × 10 crossbar array were analyzed in [19]. Operation of one-hidden layer perceptron classifier entirely in the mixed-signal integrated hardware was demonstrated in [20]. Therefore, the research on memristor neural networks is very necessary and meaningful. Although many papers have extended the memristor neural networks and solved some problems, there are still problems in the memristor neural networks. Therefore, memristor neural networks including their various kinds of deformation have broad market prospects. Especially, the research on memristor neural networks with model uncertainties has become a hot topic.

In recent decades, scholars have carried out a great amount of research and analysis on memristor neural networks. The results can be broadly divided into four categories: (1) Stability analysis of memristor neural networks [21–25]; (2) State estimation of memristor neural networks [26–29]; (3) Synchronization problem of memristor neural networks [30–32]; (4) Control problem of memristor neural networks [33–35]. In practice, time-varying delays must exist in the hardware implementation of memristor neural networks. Due to the existence of time-varying delays, the future states of the system are affected by the previous states, which leads to instability of the system and poor control performance. Consequently, state estimation of memristor neural networks is of great research value and a large part of the research has focused on state estimation of memristor neural networks. Note that the above results are generally based on the known structures and parameters of memristor neural networks without model uncertainties. In practice, the hardware implementation of memristor neural networks usually fails to attain the ideal design values, and there are design deviations. In particular, model uncertainties often exist in the hardware implementation of memristor neural networks. Therefore, model uncertainties and model errors are common in hardware memristor neural networks. Similarly, affected by model uncertainties, state estimation of memristor neural networks is also a challenging problem. Considering the above analysis, it is needed to study state estimation of memristor neural networks with model uncertainties.

A great amount of valuable research on state estimation of memristor neural networks with model uncertainties can be found in [26–29,36–38]. In [26], it used passivity theory to deal with the state estimation problem of memristor-based recurrent neural networks with time-varying delays. By using Lyapunov–Krasovskii function (LKF), convex combination technique and reciprocal convexity technique, a delay-dependent state estimation matrix was established, and the expected estimator gain matrix was obtained by solving linear matrix inequalities (LMIs). It is a pity that the model of the system must be determined and the functions in the system must be known. In [27], for memristor neural networks with randomness, the random system was transformed into an interval parameter system by Filippov, and the H∞ state observer was designed on this basis. One of the problems in the paper is that it is a random interference that affects the system rather than model uncertainty. The random interference is regular and limited. In [28], for memristor-based bidirectional associative memory neural networks with additive time-varying delays, a state estimation matrix was constructed by selecting an appropriate LKF and using the Cauchy-Schwartz-based summation inequality, and the gain matrix was obtained by the LMIs. The paper also has the problems mentioned above. In [29], for a class of memristor neural networks with different types of inductance functions and uncertain time-varying delays, a state estimation matrix was constructed by selecting a suitable LKF, and the gain matrix was solved by using the LMIs and Wirtinger-type inequality. Model uncertainty is involved in the paper, but it is only for the uncertainty of the time-varying delays. In [36], an extended dissipative state observer was proposed by using nonsmooth analysis and a new LKF. In [37], based on the basic properties of quaternion-valued, a state observer was designed for quaternion-valued memristor neural networks, and algebraic conditions were given to ensure global dissipation. The methods proposed in [36,37] are not suitable for memristor neural networks with model uncertainties. In [38], for memristor neural networks with random sampling, the randomness was represented by two different sampling periods, which satisfied a Bernoulli distribution. The random sampling system was transformed into a system with random parameters by using an input delay method. On this basis, a state observer was designed based on the LMIs and a LKF. Through the above discussion, it is not difficult to find that a similar method is used to estimate the states of memristor neural networks. By selecting an appropriate LKF, the state observation matrix is constructed based on the structure of the system, and the gain matrix is solved by utilizing the LMIs. It can be seen from the above analysis that most studies on state estimation of memristor neural networks have the same problem, which requires that the system cannot contain the model uncertainties. Some studies include model uncertainties, which are only for time-varying delays. Other studies also include model uncertainties, which are only about the fluctuation of parameters. There are few studies on the state estimation of memristor neural networks whose model uncertainties include time-varying delays, floating parameters and unknown functions. It has a huge research potential to tap.

When the memristor neural networks are designed and translated into hardware by the designer, the model uncertainties of the system only include the time-varying delays and floating parameters. In practice, the situation is not unique. Sometimes it is necessary to analyze the memristor neural networks designed by other designers. At this time, the model uncertainties of memristor neural networks include time-varying delays, floating parameters and unknown functions. The model of the memristor neural networks can be designed as in Figure 1 [28]. Motivated by the above discussion, the main concern of this paper is to design a state observer for memristor neural networks with model uncertainties, which include time-varying delays, floating parameters and unknown functions. Model uncertainties are composed of current states, past states and unknown functions. In order to approach the model uncertainties that contain memory information, improved long short term memory neural networks (LSTMs) are proposed. It is theoretically proved that the improved LSTMs can approach the model uncertainties with arbitrary error. Memristor neural networks with model uncertainties can be transformed into a new system with an improved LSTMs. On this basis, a full-order state observer is designed according to the output of the system. An error matrix of the states is constructed by a designed LKF, and the gain matrix is solved by the LMIs. In order to make the new system more accurate, a new error matrix of the states is constructed by using Young's inequality based on a LKF. On this basis, adaptive updating laws of the weights of improved LSTMs are designed to reduce the errors of the states. The main contributions of this paper are as follows.

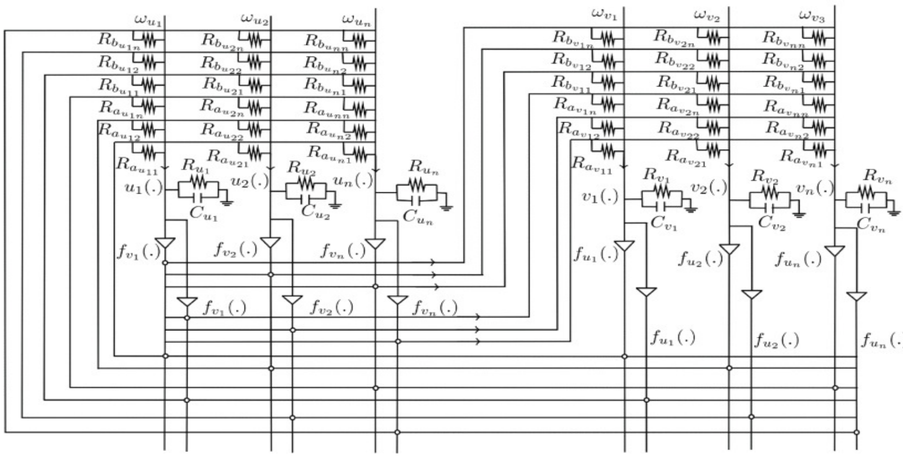

Figure 1. Circuit of memristor neural networks [28].

1. Improved LSTMs are proposed for memristor neural networks with model uncertainties. It is proved that the improved LSTMs can well approach the model uncertainties in memristor neural networks. Model uncertainties include time-varying delays, floating parameters and unknown functions. It has not been seen in other studies.
2. By utilizing the LMIs and a LKF, a full-order observer based on the output of the system is presented to obtain state information and solve the problem of state estimation.
3. By using Young's inequality and a designed LKF, adaptive updating laws of the weights of improved LSTMs are given to obtain the new system with improved LSTMs precisely.

This paper is organized as follows. In Section 2, the problem is formulated, and several essential assumptions and lemmas are listed. Section 3 presents the primary theorems, including improved LSTMs, observer design for memristor neural networks with model uncertainties, and adaptive updating laws of the weights of improved LSTMs. In Section 4, the effectiveness of the proposed scheme is demonstrated through numerical examples. Finally, the conclusions are drawn in Section 5.

Notation: $\mathcal{R}^n$ denotes the n dimensional Euclidean space. For a given matrix $\mathbf{A}$ or vector $\mathbf{B}$, $\mathbf{A}^{\mathrm{T}}$ and $\mathbf{B}^{\mathrm{T}}$ denote their transpose, and $tr\{\mathbf{A}\}$ denotes its trace. $\mathbf{A} < 0$ indicate a negative definite matrix.

2. Preliminaries

Considering the memristor neural networks as follows, the same model can be found in [26–28,36,37],

$$\begin{cases} \dot{x}_1(t) = -a_1 x_1(t) + \sum\limits_{j=1}^{n} b_{1j}(x_1(t)) f_j(x_j(t)) + \sum\limits_{j=1}^{n} c_{1j}(x_1(t)) g_j(x_j(t-\tau_j(t))) + U_1 \\ \dot{x}_2(t) = -a_2 x_2(t) + \sum\limits_{j=1}^{n} b_{2j}(x_2(t)) f_j(x_j(t)) + \sum\limits_{j=1}^{n} c_{2j}(x_2(t)) g_j(x_j(t-\tau_j(t))) + U_2 \\ \quad \vdots \\ \dot{x}_n(t) = -a_n x_n(t) + \sum\limits_{j=1}^{n} b_{nj}(x_n(t)) f_j(x_j(t)) + \sum\limits_{j=1}^{n} c_{nj}(x_n(t)) g_j(x_j(t-\tau_j(t))) + U_n \\ y_1(t) = \sum\limits_{j=1}^{n} h_{1j} x_j(t) \\ y_2(t) = \sum\limits_{j=1}^{n} h_{2j} x_j(t) \\ \quad \vdots \\ y_m(t) = \sum\limits_{j=1}^{n} h_{nj} x_j(t) \end{cases}, \tag{1}$$

where $x_i(t)(i = 1, \cdots, n)$ represents the state variable of the memristor neural networks, and n is the system dimension; $a_i(i = 1, \cdots, n)$ is the self-feedback coefficient, which satisfies $a_i > 0$; $f_j(x_j(t))$ and $g_j(x_j(t - \tau_j(t)))$ $(j = 1, \cdots, n)$ represent the activation functions of states $x_j(t)$ and $x_j(t - \tau_j(t)$ respectively; $b_{ij}(x_i(t))$ represents the memristive synaptic connection weight between states $x_i(t)$ and $x_j(t)$, and $c_{ij}(x_i(t))$ represents the memristive synaptic connection weight between states $x_i(t)$ and $x_j(t - \tau_j(t))$; $\tau_j(j = 1, \cdots, n)$ denotes the time-varying delay which satisfies $0 \leqslant \tau_j \leqslant \tau_{max}$, and τ_{max} is the upper bound constant; $U_i(i = 1, \cdots, n)$ denotes the input of the system, and $y_i(t)(i = 1, \cdots, m)$ represents the measurement output of the system; $h_{ij}(i = 1, \cdots, m; j = 1, \cdots, n)$ is the measurement constant from state $x_j(t)$ to output $y_i(t)$, and m is the output dimension.

The system (1) can be represented in vector form,

$$\begin{cases} \dot{\mathbf{x}}(t) = -\mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{f}(\mathbf{x}(t)) + \mathbf{C}\mathbf{g}(\mathbf{x}(t-\tau(t))) + \mathbf{U} \\ \mathbf{y}(t) = \mathbf{H}\mathbf{x}(t) \end{cases}, \tag{2}$$

where $\mathbf{A} = \begin{bmatrix} a_1 & 0 & \cdots & 0 \\ 0 & a_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \end{bmatrix}$, $\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nn} \end{bmatrix}$, $\mathbf{U} = \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_n \end{bmatrix}$,

$\mathbf{f}(\mathbf{x}(t)) = \begin{bmatrix} f_1(x_1(t)) \\ f_2(x_2(t)) \\ \vdots \\ f_n(x_n(t)) \end{bmatrix}$, $\mathbf{g}(\mathbf{x}(t-\tau(t))) = \begin{bmatrix} g_1(x_1(t-\tau_1(t))) \\ g_2(x_2(t-\tau_2(t))) \\ \vdots \\ g_n(x_n(t-\tau_n(t))) \end{bmatrix}$, $\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_m(t) \end{bmatrix}$,

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix}, \mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{m1} & c_{m2} & \cdots & h_{mn} \end{bmatrix}, \mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}.$$

As mentioned in the introduction, most studies involve model uncertainties that only include floating parameters. In the process of neural network hardware implementation as memristor neural networks, the memristive synaptic connection weights b_{ij} and c_{ij} will produce deviations [28]. The fluctuation of parameters b_{ij} and c_{ij} is regarded as model uncertainty. This is the starting point of much research on state estimation of memristor neural networks, such as [26–28,36–38]. Some studies regard time-varying delay $\tau_j(t)$ as model uncertainty and study state estimation of memristor neural networks based on it, for example [29]. It should be noted that model uncertainties in all the above studies do not include $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$. Both $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$ must be known, and b_{ij} and c_{ij} float within the ideal range. If $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$ are unknown, and the ideal values of b_{ij} and c_{ij} are unknown, the model uncertainties include floating parameters b_{ij} and c_{ij}, time-varying delay $\tau_j(t)$ and unknown functions $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$, and all the above studies are not applicable. The state estimation of memristor neural networks with model uncertainties including floating parameters, time-varying delays and unknown functions is the main concern in this paper.

Remark 1. *In other studies, model uncertainties only include floating parameters b_{ij} and c_{ij} or time-varying delay $\tau_j(t)$. Functions $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$ must be known. In this paper, model uncertainties include floating parameters b_{ij} and c_{ij}, time-varying delay $\tau_j(t)$ and unknown functions $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$.*

As shown in system (2), the model uncertainties contain memory information $x_i(t-\tau_i)$. The LSTMs are the most suitable to deal with the model uncertainties. LSTMs are networks of basic LSTMs cells, and the architecture of a conventional LSTMs cell is illustrated in Figure 2. A memory cell, an input gate, an output gate and a forgetting gate make up a LSTMs cell. The forgetting gate, input gate, and output gate respectively determine whether historical information, input information, and output information are retained [39]. The specific computation is shown in Equation (3).

$$\begin{aligned}
\begin{pmatrix} \mathbf{f}_t \\ \mathbf{i}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} &= \begin{pmatrix} \sigma\left(\mathbf{W}_f[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_f\right) \\ \sigma(\mathbf{W}_i[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_i) \\ \sigma(\mathbf{W}_o[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_o) \\ tanh(\mathbf{W}_g[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_g) \end{pmatrix} \\
\mathbf{c}_t &= \mathbf{f}_t \otimes \mathbf{c}_{t-1} \oplus \mathbf{i}_t \otimes \mathbf{g}_t \\
\mathbf{h}_t &= \mathbf{o}_t \otimes tanh(\mathbf{c}_t),
\end{aligned} \tag{3}$$

where $\mathbf{f}_t$ denotes the forgetting gate; $\mathbf{i}_t$ and $\mathbf{o}_t$ represent input gate and output gate, respectively; $\mathbf{g}_t$ is the updating vector of the LSTM cell; $\mathbf{h}_t$ is the hidden state vector; $\mathbf{h}_{t-1}$ is the hidden state vector at step $t-1$; $\mathbf{x}_t$ is the input vector of the LSTM cell; $\mathbf{c}_t$ is the state vector of the cell; $\mathbf{c}_{t-1}$ is the state vector of the cell at step $t-1$; $\mathbf{W}$ is the weight matrix and $\mathbf{b}$ refers to the bias vector; σ and $tanh(\cdot)$ are the sigmoid and tanh activation functions, respectively; $\otimes$ and $\oplus$ represent elementwise multiplication and addition, respectively.

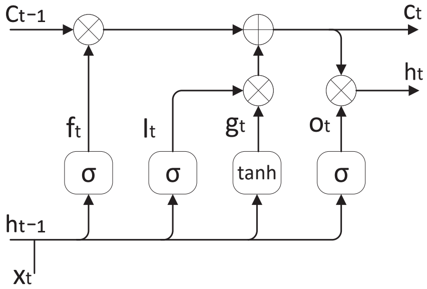

Figure 2. Schematic diagram of a basic LSTMs cell.

Remark 2. *LSTMs cell is not completely suitable for estimating the states of memristor neural networks with model uncertainties. LSTMs cell needs to be improved to save computation and be more suitable for state estimation.*

Moreover, in order to improve the LSTMs, design the state observer of memristor neural networks with model uncertainties, and derive the updating laws of the weights of the improved LSTMs, some assumptions and lemmas need to be introduced for the following proof.

Assumption 1. *The functions $f_j(\cdot)$ and $g_j(\cdot)$ satisfy local Lipschitz conditions. For all $k, p \in \mathcal{R}$, have $\left|f_j(k) - f_j(p)\right| \leqslant K_f|k-p|$ and $\left|g_j(k) - g_j(p)\right| \leqslant K_g|k-p|$, where K_f and K_g are Lipschitz constants, and satisfy $f_j(0) = g_j(0) = 0$.*

$f_j(\cdot)$ and $g_j(\cdot)$ are the activation functions of memristor neural networks, so Assumption 1 is generally tenable.

Lemma 1 ([40]). *$k(\cdot)$ is a continuous function defined on a set Ω. Multilayer neural networks can be defined as,*

$$\bar{k} = \mathbf{W}^{\mathrm{T}} S(\mathbf{VI}),$$

*where **W** and **V** are the second weight matrix and the first weight vector of the Multilayer neural networks, respectively; **I** is the input vector of Multilayer neural networks, and $S(\cdot)$ is the activation function of Multilayer neural networks.*
Then, for a given desired level of accuracy $\varepsilon > 0$, there exist the ideal weights $\bar{\mathbf{W}}$ and $\bar{\mathbf{V}}$ to satisfy the following inequality,

$$\sup_{\mathbf{I}\in\Omega} \|k(\cdot) - \bar{k}\| \leqslant \varepsilon.$$

Lemma 2. *(Young's inequality) For all $x, y \in \mathcal{R}$, the following inequality holds,*

$$xy \leqslant \frac{\varepsilon^p}{p}\|x\|^p + \frac{1}{q\varepsilon^p}\|y\|^q,$$

where $\varepsilon > 0$, $p > 1$, $q > 1$, and $(p-1)(q-1) = 1$.

3. Main Result

In this part, improved LSTMs, state observer design for memristor neural networks with model uncertainties, and adaptive updating laws of the weights of improved LSTMs will be discussed.

To begin with, the system (2) can be redefined as follows,

$$\begin{cases} \dot{\mathbf{x}}(t) = -\mathbf{A}\mathbf{x}(t) + \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))) + \mathbf{U} \\ \mathbf{y}(t) = \mathbf{H}\mathbf{x}(t) \end{cases}, \tag{4}$$

where $\mathbf{K}(\cdot)$ is a vector of functions, which can be defined as $[K_1(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))), K_2(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))), \cdots, K_n(\mathbf{x}(t), \mathbf{x}(t-\tau(t))))]^{\mathrm{T}}$.

As mentioned in Remark 1, $\mathbf{K}(\cdot)$ is the function vector of model uncertainties formed by floating parameters b_{ij} and c_{ij}, time-varying delay $\tau_j(t)$ and unknown functions $f_i(x_i(t))$ and $g_i(x_i(t-\tau_i))$. In order to approximate the unknown function vector $\mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t))))$, improved LSTMs are proposed, and an improved LSTMs cell is shown in Figure 3.

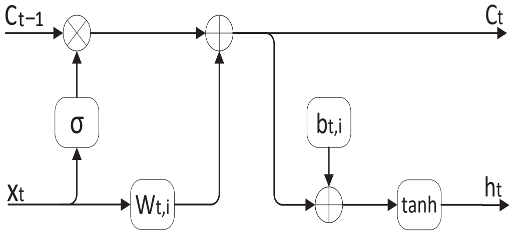

Figure 3. Schematic diagram of an improved LSTMs cell.

Comparing Figures 2 and 3, it can be seen that the input gate $\mathbf{i}_t$ and the hidden state vector $\mathbf{h}_{t-1}$ at step $t-1$ have been removed. Since $\mathbf{x}(t)$ is part of $\mathbf{K}(\cdot)$ in the form of a function vector, the input gate can be removed. $\mathbf{x}(t)$ should be part of LSTMs cell in the form of tanh function. The reason why $\mathbf{h}_{t-1}$ is removed is that $\mathbf{K}(\cdot)$ contains $\mathbf{x}(t-\tau(t))$, so the functions of $\mathbf{h}_{t-1}$ can be combined into $\mathbf{c}_{t-1}$ to save computation. Remove the output gate $\mathbf{o}_t$ and use $\mathbf{h}_t$ as the output of LSTMs cell to simplify the structure of LSTMs cell. Therefore, the improved LSTMs cell is made up of the following parts: (1) The state vector $\mathbf{x}(t)$ of the system at time t and the state vector with weights $\mathbf{c}_{t-1}$ of the system at time $t-1$ constitute the input of the improved LSTMs cell; (2) $\mathbf{c}_t$ is the vector that holds the state of the improved LSTMs cell at time t; (3) $\mathbf{h}_t$ is the output vector of the improved LSTMs cell at time t; (4) $\sigma(\mathbf{x}(t))$ is the forgetting function at time t, which is used to control whether the memory information stored by the improved LSTMs cell at time $t-1$ is added to the improved LSTMs cell calculation at time t. The specific computation of a simplified and improved LSTMs cell can be expressed as follows,

$$\begin{aligned} \mathbf{c}_t &= \mathbf{W}_{t,i}\mathbf{x}_t \oplus \mathbf{c}_{t-1} \otimes \sigma(\mathbf{x}_t) \\ \mathbf{h}_t &= tanh(\mathbf{W}_{t,i}\mathbf{x}_t \oplus \mathbf{c}_{t-1} \otimes \sigma(\mathbf{x}_t) \oplus b_{t,i}), \end{aligned} \tag{5}$$

where $\mathbf{W}_{t,i} = [W_{t,i,1}, W_{t,i,2}, \cdots, W_{t,i,n}]$ denotes the weight vector of the ith cell and $b_{t,i}$ is a bias constant of the ith cell; $\mathbf{x}_t = [x_1(t), x_2(t), \cdots, x_n(t)]^{\mathrm{T}}$ represents the state vector at time t.

Based on the simplified and improved LSTMs cell, the improved LSTMs are illustrated in Figure 4. In Figure 4, each column represents a neural network composed of p improved LSTMs cells at time j. The outputs of the p improved LSTMs cells pass through the weight matrix $\mathbf{V}_j$ to obtain the output vector of the neural network at time j, which is used to approximate $\mathbf{K}(\cdot)$. The neural network at each time can be connected through $\mathbf{c}_j$ and $\mathbf{c}_{j-1}$ to form neural networks at all times. x_j represents the state vector at time j, and $j \in [1, t]$. c_j^i denotes the output of the hidden states of the ith$(i \in [1, p])$ LSTMs cell at time j, and p is the number of LSTMs cells. $\mathbf{W}_{j,i}$ represents the weight vector of the ith LSTMs cell at time j. $b_{j,i}$ is the bias of the ith LSTMs cell at time j. h_j^i denotes the output of the states of the ith LSTMs cell at time j. $V_{j,i,l}$ represents the weight coefficient from the output of the ith LSTMs cell to the lth system output at time j, and $l \in [1, m]$. $y_{j,l}$ denotes the lth system output at time j. The improved LSTMs can approximate any nonlinear function by the following theorem.

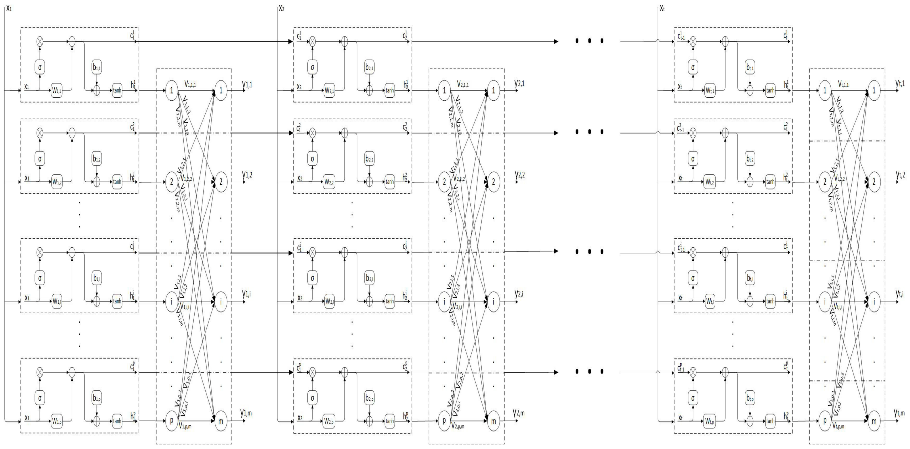

Figure 4. Schematic diagram of the improved LSTMs.

Theorem 1. *$k(\cdot)$ is a continuous nonlinear function defined on a set Ω. Improved LSTMs are shown in Figure 3. $\bar{k}$ is an approximate function of $k(\cdot)$ based on the improved LSTMs. Then, for a given desired level of accuracy $\varepsilon > 0$, there exist the ideal weights $\bar{\mathbf{W}}_{j,i}(j \in [1,t], i \in [1,p])$ and $\bar{V}_{j,i,l}(j \in [1,t], i \in [1,p], l \in [1,m])$ to satisfy the following inequality,*

$$\sup_{\mathbf{x}_j \in \Omega} \|k(\cdot) - \bar{k}\| \leqslant \varepsilon. \tag{6}$$

The proof of Theorem 1 can be found in Appendix A.

Based on Theorem 1, the estimation system for the system (4) can be defined as the following formula,

$$\begin{cases} \dot{\hat{\mathbf{x}}}(t) = -\mathbf{A}\hat{\mathbf{x}}(t) + \bar{\mathbf{K}} + \mathbf{L} \cdot [\mathbf{y}(t) - \mathbf{H} \cdot \hat{\mathbf{x}}(t)] + \mathbf{U} \\ \hat{\mathbf{y}}(t) = \mathbf{H}\hat{\mathbf{x}}(t) \end{cases}, \tag{7}$$

where $\mathbf{L} \in \mathcal{R}^{n \times m}$ denotes the observer gain matrix; $\bar{\mathbf{K}} \in \mathcal{R}^{n}$ is an estimated function vector of $\mathbf{K}(\mathbf{x}(t), \mathbf{x}(t - \tau(t))))$ based on the improved LSTMs, which satisfies Theorem 1. $\bar{\mathbf{K}}$ is given in Equation (8),

$$\bar{\mathbf{K}} = \bar{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t \mathbf{x}_t + \sum_{i=1}^{t-1} \left[\bar{\mathbf{W}}_i \mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j) \right] + \bar{\mathbf{b}}_t \right\}, \tag{8}$$

where $\bar{\mathbf{V}}_t \in \mathcal{R}^{p \times n}$ and $\bar{\mathbf{W}}_i \in \mathcal{R}^{p \times n}$ denote the ideal weight matrices, and $\bar{\mathbf{b}}_t \in \mathcal{R}^{p}$ is the ideal bias vector.

The function $\sigma(\mathbf{x}_j)$ is determined by the time-varying delay $\tau_j(t)$ which satisfies $0 \leqslant \tau_j(t) \leqslant \tau_{max}$. $\sigma(\mathbf{x}_j)$ is 1 in the range of $[t - \tau_{max}, t]$, and $\sigma(\mathbf{x}_j)$ is 0 in the rest of the range. This ensures that all the data in the interval $t - \tau_{max}$ to t will be included in the calculation. Considering the system (4) and the estimation system (7), the error system can be obtained as follows

$$\begin{aligned} \mathbf{e}(t) &= \hat{\mathbf{x}}(t) - \mathbf{x}(t), \\ \dot{\mathbf{e}}(t) &= \dot{\hat{\mathbf{x}}}(t) - \dot{\mathbf{x}}(t) \\ &= [-\mathbf{A}\hat{\mathbf{x}}(t) + \bar{\mathbf{K}} + \mathbf{L} \cdot [\mathbf{y}(t) - \mathbf{H} \cdot \hat{\mathbf{x}}(t)] + \mathbf{U}] \\ &\quad - [-\mathbf{A}\mathbf{x}(t) + \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t - \tau(t)))) + \mathbf{U}] \\ &= -(\mathbf{A} + \mathbf{L}\mathbf{H})\mathbf{e}(t) + [\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t - \tau(t)))]. \end{aligned} \tag{9}$$

Assumption 2. *For the unknown function $K_i(\mathbf{x}(t), \mathbf{x}(t-\tau(t))))$ and the estimated function $\bar{K}_i$ $(i = 1, 2, \cdots, n)$, there exist Lipschitz constant vectors $\mathbf{K}_{L1}$ and $\mathbf{K}_{L2}$, which satisfy the following inequality,*

$$|K_i(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))) - \bar{K}_i| \leqslant \mathbf{K}_{L1}^{\mathrm{T}}|\mathbf{x}(t) - \hat{\mathbf{x}}(t)| + \mathbf{K}_{L2}^{\mathrm{T}}|\mathbf{x}(t-\tau(t)) - \hat{\mathbf{x}}(t-\tau(t))|. \tag{10}$$

Considering Theorem 1, $\bar{K}_i$ is an estimated function of the finite error of $K_i(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))$. Similarly, $\bar{K}_i$ is a function of $\hat{\mathbf{x}}(t)$ and $\hat{\mathbf{x}}(t-\tau(t))$. On this basis, considering Assumption 1, Assumption 2 is tenable.

Theorem 2. *Suppose that Assumption 2 holds for the system (4) and the estimation system (7), if there exist symmetric positive definite matrices $\mathbf{P}$, $\mathbf{Q}$, $\mathbf{M}$, a diagonal matrix $\mathbf{F}$, a matrix $\mathbf{G} \in \mathcal{R}^{n\times p}$ and a real constant $\delta > 0$ such that inequality (11) holds,*

$$\begin{bmatrix} \mathbf{\Omega}_1 & \mathbf{P} & \mathbf{M} & \mathbf{M} \\ \mathbf{P} & -\mathbf{F} & 0 & 0 \\ \mathbf{M} & 0 & \mathbf{\Omega}_2 & -2\mathbf{M} \\ \mathbf{M} & 0 & -2\mathbf{M} & -2\mathbf{M} \end{bmatrix} < 0, \tag{11}$$

where $\mathbf{\Omega}_1 = -\mathbf{A}^{\mathrm{T}}\mathbf{P} - \mathbf{H}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}} - \mathbf{PA} - \mathbf{GH} + \mathbf{Q} + 2\mathbf{K}_{L1}tr\{\mathbf{F}\}\mathbf{K}_{L1}^{\mathrm{T}}$, *and* $\mathbf{\Omega}_2 = 2\mathbf{K}_{L2}tr\{\mathbf{F}\}\mathbf{K}_{L2}^{\mathrm{T}} - (1-\delta)\mathbf{Q} - 2\mathbf{M}$.

Then, the error system (9) is asymptotically stable with observer gain matrix calculated by $\mathbf{L} = \mathbf{P}^{-1}\mathbf{G}$. *The proof of Theorem 2 can be found in Appendix B.*

Based on Theorem 2, the observer gain matrix $\mathbf{L}$ can be obtained. Considering the function vector $\bar{\mathbf{K}}$ in system (7), the weight matrices $\bar{\mathbf{W}}_i$ and $\bar{\mathbf{V}}_i$ are ideal. In fact, the ideal weights are hard to select, and the estimated weights need to be adjusted by adaptive laws to be close to the ideal weights. With reference to the system (7), the estimated system can be redefined as follows

$$\begin{cases} \dot{\hat{\mathbf{x}}}(t) = -\mathbf{A}\hat{\mathbf{x}}(t) + \hat{\mathbf{K}} + \mathbf{L}\cdot[\mathbf{y}(t) - \mathbf{H}\cdot\hat{\mathbf{x}}(t)] + \mathbf{U} \\ \hat{\mathbf{y}}(t) = \mathbf{H}\hat{\mathbf{x}}(t) \end{cases}, \tag{12}$$

where $\hat{\mathbf{K}}$ is an estimated function vector of $\bar{\mathbf{K}}$.

$\hat{\mathbf{K}}$ is given in Equation (13),

$$\hat{\mathbf{K}} = \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{\hat{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\hat{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \hat{\mathbf{b}}_t\right\}, \tag{13}$$

where $\hat{\mathbf{V}}_t$ and $\hat{\mathbf{W}}_i$ are estimated weight matrices; $\hat{\mathbf{b}}_t$ is a estimated bias vector.

With reference to the error system (9), the error system can be obtained as follows by using the Equation (6)

$$\dot{\mathbf{e}}(t) = -(\mathbf{A} + \mathbf{LH})\mathbf{e}(t) + \left(\hat{\mathbf{K}} - \bar{\mathbf{K}} - \epsilon_1\right), \tag{14}$$

where ϵ_1 is an error vector.

For error weight matrices $\tilde{\mathbf{V}}_t$ and $\tilde{\mathbf{W}}_i$ and a error weight vector $\tilde{\mathbf{b}}_t$, we have

$$\begin{aligned} \tilde{\mathbf{V}}_t &= \hat{\mathbf{V}}_t - \bar{\mathbf{V}}_t \\ \tilde{\mathbf{W}}_i &= \hat{\mathbf{W}}_i - \bar{\mathbf{W}}_i \\ \tilde{\mathbf{b}}_t &= \hat{\mathbf{b}}_t - \bar{\mathbf{b}}_t. \end{aligned} \tag{15}$$

Theorem 3. *For the error system (14), the design parameters* $\mathbf{N}_{w\_i} \in \mathcal{R}^p$*,* $\mathbf{N}_{v\_t} \in \mathcal{R}^p$ *and* $\mathbf{N}_{b\_t} \in \mathcal{R}^{p\times n}$ *satisfy following inequality*

$$\begin{bmatrix} \mathbf{\Omega}_3 & -\mathbf{P} & \frac{1}{2}\mathbf{\Omega}_{wvb}^{\mathrm{T}} \\ -\mathbf{P} & \mathbf{0} & \mathbf{0} \\ \frac{1}{2}\mathbf{\Omega}_{wvb} & \mathbf{0} & \mathbf{0} \end{bmatrix} \leqslant 0,$$

where $\mathbf{\Omega}_3 = -(\mathbf{A}+\mathbf{LH})^{\mathrm{T}}\mathbf{P} - \mathbf{P}(\mathbf{A}+\mathbf{LH})$*,*

$$\mathbf{\Omega}_{wvb} = \begin{bmatrix} \mathbf{\Omega}_{w\_t} \\ \mathbf{\Omega}_{w\_t-1}\cdot\sigma(\mathbf{x}_t) \\ \vdots \\ \mathbf{\Omega}_{w\_1}\cdot\prod_{j=2}^{t}\sigma(\mathbf{x}_j) \\ \mathbf{\Omega}_{v\_t} \\ \mathbf{N}_{b\_t} \end{bmatrix}, \mathbf{\Omega}_{w\_i} = \begin{bmatrix} \mathbf{N}_{w\_i} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_{w\_i} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{N}_{w\_i} \end{bmatrix},$$

$$\mathbf{\Omega}_{v\_t} = \left(\begin{bmatrix} \mathbf{N}_{v\_t} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_{v\_t} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{N}_{v\_t} \end{bmatrix} - 2\begin{bmatrix} \epsilon_2\cdot\mathbf{P}_{1\bullet} \\ \epsilon_2\cdot\mathbf{P}_{2\bullet} \\ \vdots \\ \epsilon_2\cdot\mathbf{P}_{n\bullet} \end{bmatrix} \right).$$

The adaptive updating laws of the weights can be given as follows

$$\begin{aligned} \dot{\hat{\mathbf{W}}}_i &= \mathbf{N}_{w\_i}\cdot\mathbf{e}^{\mathrm{T}}(t) - 4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t)\mathbf{x}_i^{\mathrm{T}} \quad (i = 1,2,\cdots,t) \\ \dot{\hat{\mathbf{V}}}_t &= \mathbf{N}_{v\_t}\cdot\mathbf{e}^{\mathrm{T}}(t) - 2\hat{\mathbf{S}}_t\mathbf{e}^{\mathrm{T}}(t)\mathbf{P} \\ \dot{\hat{\mathbf{b}}}_t &= \mathbf{N}_{b\_t}\cdot\mathbf{e}(t) - 4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t), \end{aligned} \tag{16}$$

then the error system (14) is asymptotically stable. The proof of Theorem 3 can be found in Appendix C.

Considering (16), the adaptive updating laws of the weights are determined by $\mathbf{e}(t)$. Hence, it requires that $\mathbf{e}(t)$ is a n-dimensional vector. According to (12), we have

$$\hat{\mathbf{y}}(t) - \mathbf{y}(t) = \mathbf{H}\cdot(\hat{\mathbf{x}}(t) - \mathbf{x}(t)) = \mathbf{H}\cdot\mathbf{e}(t). \tag{17}$$

If there exists $\mathbf{H}^{-1}$, the $\mathbf{e}(t)$ can be obtained as follows by using (17)

$$\mathbf{e}(t) = \mathbf{H}^{-1}\cdot(\hat{\mathbf{y}}(t) - \mathbf{y}(t)). \tag{18}$$

In general, m is not equal to n, and $\mathbf{H}^{-1}$ does not exist. Hence, (18) does not hold. To solve this problem, the following assumption is given.

Assumption 3. $\hat{\mathbf{y}}(t)$ *and* $\mathbf{y}(t)$ *are continuously differentiable functions and the first derivative of* $\hat{\mathbf{y}}(t)$ *and* $\mathbf{y}(t)$ *are bounded and measurable.*

Theorem 4. *Based on Assumption 3, if the given matrix* $\mathbf{G}$ *is left invertible, the* $\mathbf{e}(t)$ *can be obtained as follows*

$$\mathbf{e}(t) = \mathbf{G}^{-1}\cdot\mathbf{Y}, \tag{19}$$

where $\mathbf{G} = \begin{bmatrix} \mathbf{H} \\ -\mathbf{H}(\mathbf{A}+\mathbf{LH}) \end{bmatrix}$ *and* $\mathbf{Y} = \begin{bmatrix} \hat{\mathbf{y}}(t) - \mathbf{y}(t) \\ \dot{\hat{\mathbf{y}}}(t) - \dot{\mathbf{y}}(t) \end{bmatrix}$*. The proof of Theorem 4 can be found in Appendix D.*

Remark 3. *Based on Theorem 1, the estimated system (12) can be given. By using Theorem 2, the observer gain matrix* **L** *can be obtained. By using Theorems 3 and 4, the adaptive updating laws of the weights can be obtained.*

4. Simulation Analysis

In this section, two numerical cases are presented to verify the rationality of the above results.

4.1. Examples

Example 1. 2-dimensional memristor neural networks are considered, and the parameters of the system (2) are given as follows,

$$\mathbf{A} = \begin{bmatrix} 2.3 & 0 \\ 0 & 2 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.31 & 0.38 \\ 0.49 & 0.32 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 0.32 & 0.19 \\ 0.39 & 0.25 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} 0.2 \\ 0.3 \end{bmatrix},$$
$$\mathbf{H} = \begin{bmatrix} 1 & 0.5 \end{bmatrix}, \quad \mathbf{f}(\mathbf{x}(t)) = \begin{bmatrix} \frac{|x_1+1|-|x_1-1|}{2} \\ \frac{|x_2+1|-|x_2-1|}{2} \end{bmatrix}, \quad \mathbf{g}(\mathbf{x}(t-\tau(t))) = \begin{bmatrix} x_1(t-\tau_1(t)) \\ x_2(t-\tau_2(t)) \end{bmatrix},$$
$$\tau_1(t) = \tau_2(t) = \frac{0.05t}{1+t}, \quad \mathbf{x}(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Based on the system (2), the estimated system (17) can be designed as follows,

$$\hat{\mathbf{x}}(0) = \begin{bmatrix} 0.2 \\ 0.3 \end{bmatrix}, \quad \mathbf{K}_{L1} = \mathbf{K}_{L2} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad \delta = 0.1.$$

By using Theorem 2 and LMIs tools, the parameters of the estimated system (17) can be obtained,

$$\mathbf{P} = \begin{bmatrix} 0.03553 & -0.00345 \\ -0.00345 & 0.03629 \end{bmatrix}, \mathbf{Q} = \begin{bmatrix} 0.03545 & -0.01504 \\ -0.01504 & 0.04070 \end{bmatrix}, \mathbf{F} = \begin{bmatrix} 0.00363 & 0 \\ 0 & 0.00379 \end{bmatrix},$$
$$\mathbf{M} = \begin{bmatrix} 0.00494 & -68.44222 \\ 68.44222 & 0.00505 \end{bmatrix}, \mathbf{L} = \begin{bmatrix} -0.88667 \\ -0.75623 \end{bmatrix}.$$

Set sampling time $T = 30s$ and sampling period $\triangle T = 0.001s$. Considering (18), set $\mathbf{x}_i = \hat{\mathbf{x}}_{i\cdot\triangle T}(i = 1, 2, \cdots, t/\triangle T)$ and $\sigma(\mathbf{x}_i) = 0(i < t/\triangle T - 30)$. Based on Theorem 3 and Theorem 4, set $\mathbf{N}_{w\_i}$ and $\mathbf{N}_{v\_t}$ and $\mathbf{N}_{b\_t}$ are negative unit vectors and matrix.

The state trajectories of the state $\mathbf{x}(t)$ and the state observer $\hat{\mathbf{x}}(\mathbf{t})$ are drawn in Figure 5. Figure 6 is drawn for the estimated error between the state $\mathbf{x}(t)$ and the state observer $\hat{\mathbf{x}}(\mathbf{t})$. In Figure 7, the trajectories of the derivative of the state $\dot{\mathbf{x}}(t)$ and the derivative of the state observer $\dot{\hat{\mathbf{x}}}(t)$ are depicted. The trajectories of the error between $\dot{\mathbf{x}}(t)$ and $\dot{\hat{\mathbf{x}}}(t)$ are given in Figure 8. In Figure 9, the output curve $y(t)$ and the estimated output curve $\hat{y}(t)$ are given. Figure 10 shows the estimated error curve between $y(t)$ and $\hat{y}(t)$.

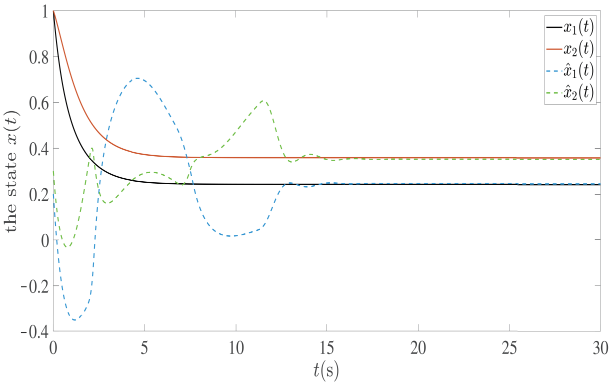

Figure 5. The state and estimated state curves of the 2-dimensional memristor neural networks.

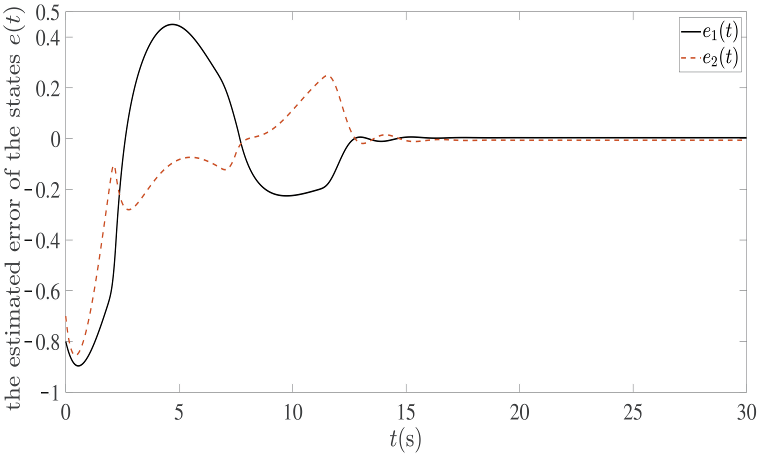

Figure 6. The estimated error curves of the states.

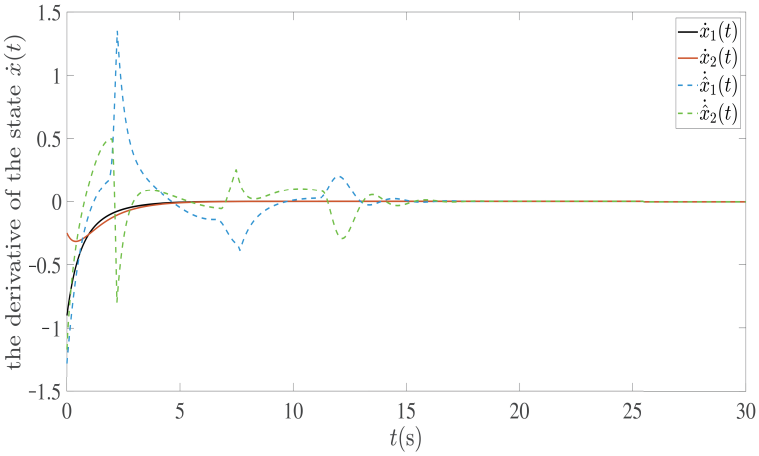

Figure 7. The derivative curves of the states and estimated states.

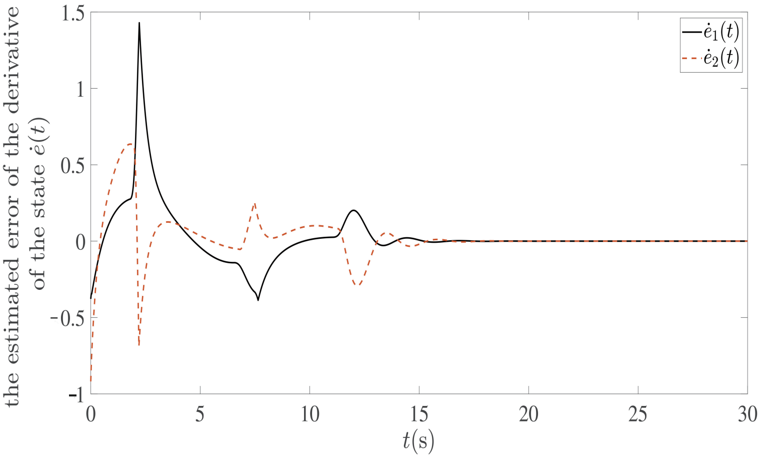

Figure 8. The error curves of the derivative of the states.

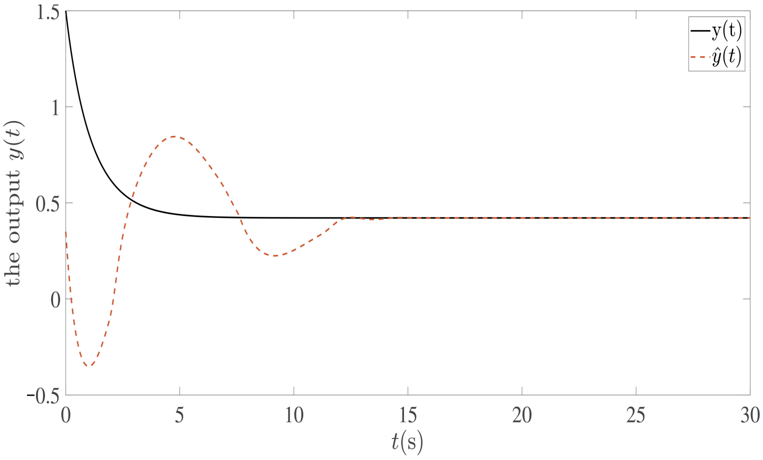

Figure 9. The output and estimated output curves.

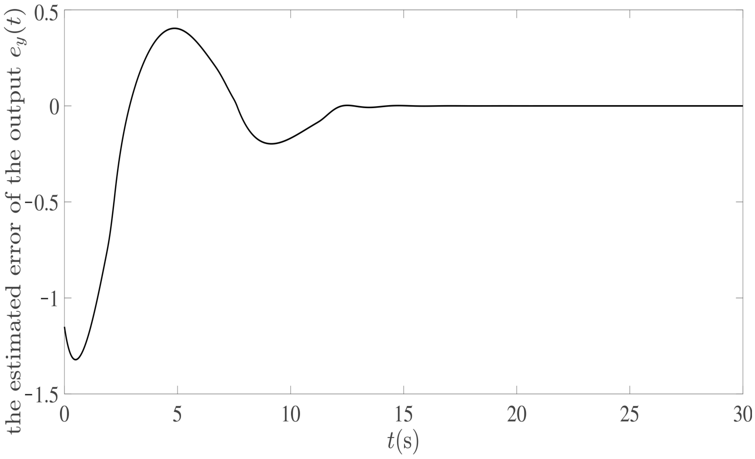

Figure 10. The error curves of the output.

In order to verify the accuracy of the estimated structure, a test system is designed based on the gain observation matrix **L**. Under the same simulation conditions as above, the effects of the adjusted weights and the random weights on the system are compared. In Figure 11, the state trajectories of the real system and the estimated system with the adjusted weights and the system with the random weights are given. Figure 12 shows the real output curve and the estimated output with the adjusted weights and the output curve with the random weights.

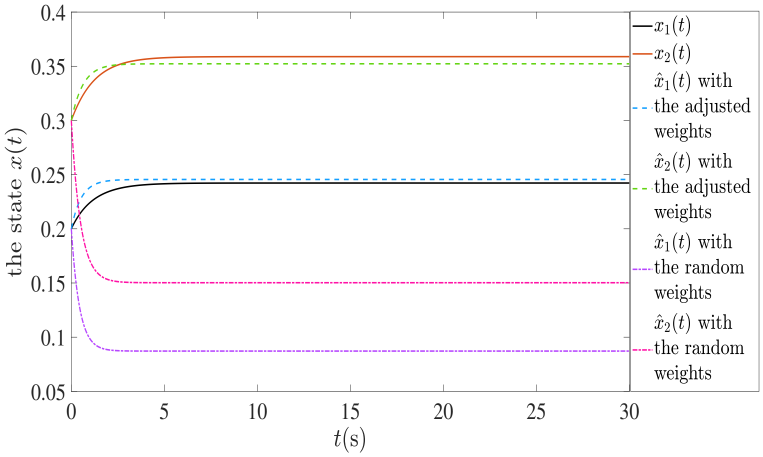

Figure 11. The state and estimated state with the adjusted weights and state with the random weights curves.

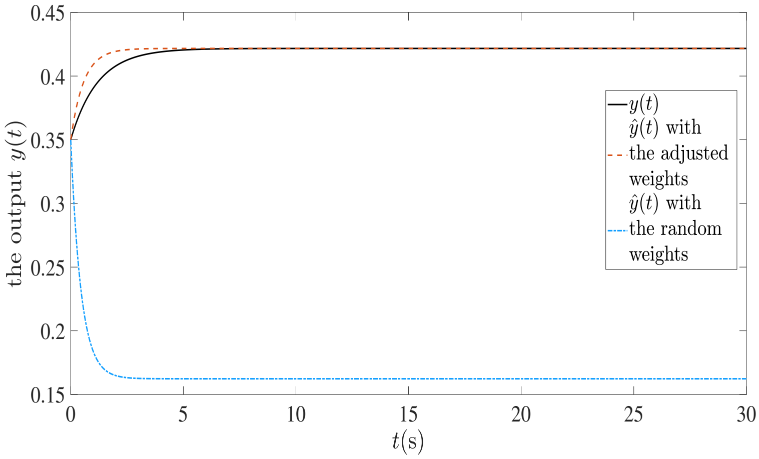

Figure 12. The output and estimated output with the adjusted weights and output with the random weights curves.

Example 2. 3-dimensional memristor neural networks are considered, and the parameters of the system (2) are given as follows,

$$\mathbf{A} = \begin{bmatrix} 2.5 & 0 & 0 \\ 0 & 2.5 & 0 \\ 0 & 0 & 3.3 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0.2 & 0.15 & 0.35 \\ 0.15 & 0.35 & 0.08 \\ 0.1 & 0.15 & 0.3 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 0.125 & 0.15 & 0.136 \\ 0.35 & 0.18 & 0.3 \\ 0.2 & 0.04 & 0.05 \end{bmatrix},$$

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{f}(\mathbf{x}(t)) = \begin{bmatrix} \frac{|x_1+1|-|x_1-1|}{2} \\ \frac{|x_2+1|-|x_2-1|}{2} \\ \frac{|x_3+1|-|x_3-1|}{2} \end{bmatrix}, \quad \mathbf{g}(\mathbf{x}(t-\tau(t))) = \begin{bmatrix} x_1(t-\tau_1(t)) \\ x_2(t-\tau_2(t)) \\ x_3(t-\tau_3(t)) \end{bmatrix},$$

$$\mathbf{U} = \begin{bmatrix} 0.2 \\ 0.3 \\ 0.1 \end{bmatrix}, \quad \mathbf{x}(0) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \tau_1(t) = \tau_2(t) = \tau_3(t) = \frac{0.05t}{1+t}.$$

Based on the system (2), the estimated system (17) can be designed as follows,

$$\hat{\mathbf{x}}(0) = \begin{bmatrix} 0.4 \\ 0.5 \\ 0.6 \end{bmatrix}, \quad \mathbf{K}_{L1} = \mathbf{K}_{L2} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \delta = 0.1.$$

By using Theorem 2 and LMIs tools, the parameters of the estimated system (17) can be obtained,

$$\mathbf{P} = \begin{bmatrix} 0.51349 & 0.03364 & 0.10199 \\ 0.03364 & 0.71344 & -0.02472 \\ 0.10199 & -0.02472 & 0.39999 \end{bmatrix}, \mathbf{Q} = \begin{bmatrix} -0.30425 & -0.32097 & -0.34837 \\ -0.32097 & -0.51236 & -0.19432 \\ -0.34837 & -0.19432 & 0.00547 \end{bmatrix},$$

$$\mathbf{F} = \begin{bmatrix} 0.10879 & 0 & 0 \\ 0 & 0.17792 & 0 \\ 0 & 0 & 0.20825 \end{bmatrix}, \quad \mathbf{M} = \begin{bmatrix} 0.05344 & -47.67758 & -16.83918 \\ 47.67685 & 0.04188 & -39.81878 \\ 16.83445 & 39.83217 & 0.07044 \end{bmatrix},$$

$$\mathbf{L} = \begin{bmatrix} -0.29468 & -0.06180 \\ -0.99352 & 0.98883 \\ 0.32407 & -0.98883 \end{bmatrix}.$$

Set sampling time $T = 30s$ and sampling period $\triangle T = 0.001s$. Considering (18), set $\mathbf{x}_i = \hat{\mathbf{x}}_{i\cdot\triangle T}(i = 1, 2, \cdots, t/\triangle T)$ and $\sigma(\mathbf{x}_i) = 0(i < t/\triangle T - 30)$. Based on Theorems 3 and 4, set $\mathbf{N}_{w\_i}$ and $\mathbf{N}_{v\_t}$ and $\mathbf{N}_{b\_t}$ are negative unit vectors and matrix.

The state trajectories of the state $\mathbf{x}(t)$ and the state observer $\hat{\mathbf{x}}(\mathbf{t})$ are drawn in Figure 13. Figure 14 is drawn for the estimated error between the state $\mathbf{x}(t)$ and the state observer

$\hat{\mathbf{x}}(\mathbf{t})$. In Figure 15, the trajectories of the derivative of the state $\dot{\mathbf{x}}(t)$ and the derivative of the state observer $\dot{\hat{\mathbf{x}}}(t)$ are depicted. The trajectories of the error between $\dot{\mathbf{x}}(t)$ and $\dot{\hat{\mathbf{x}}}(t)$ are given in Figure 16. In Figure 17, the output curve $y(t)$ and the estimated output curve $\hat{y}(t)$ are given. Figure 18 shows the estimated error curve between $y(t)$ and $\hat{y}(t)$.

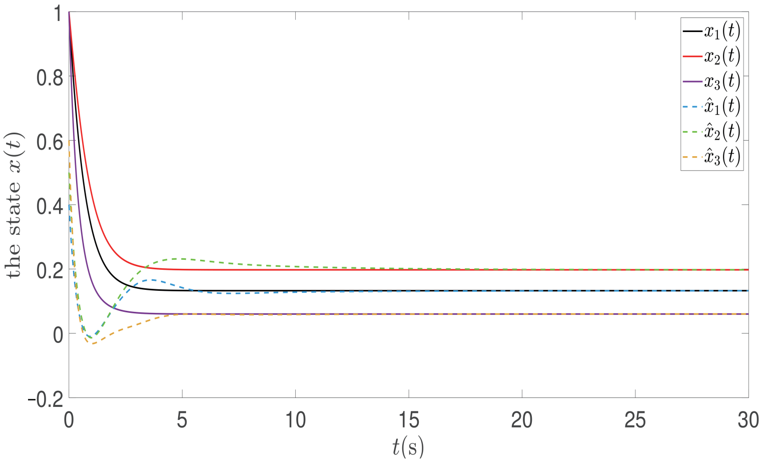

Figure 13. The state and estimated state curves of the 3-dimensional memristor neural networks.

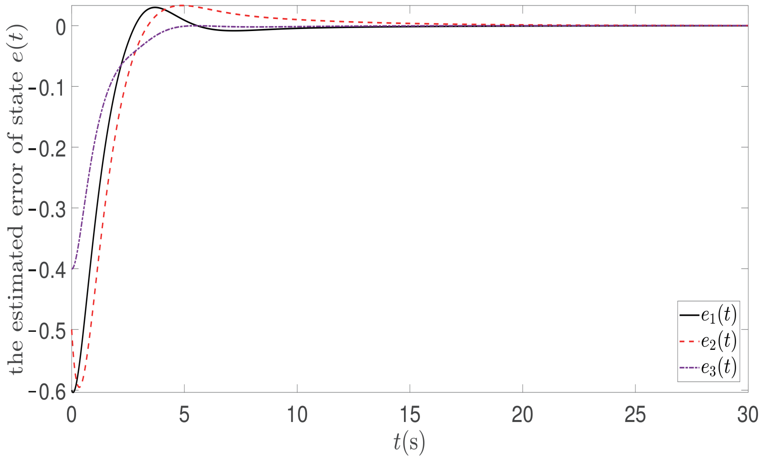

Figure 14. The estimated error curves of the states.

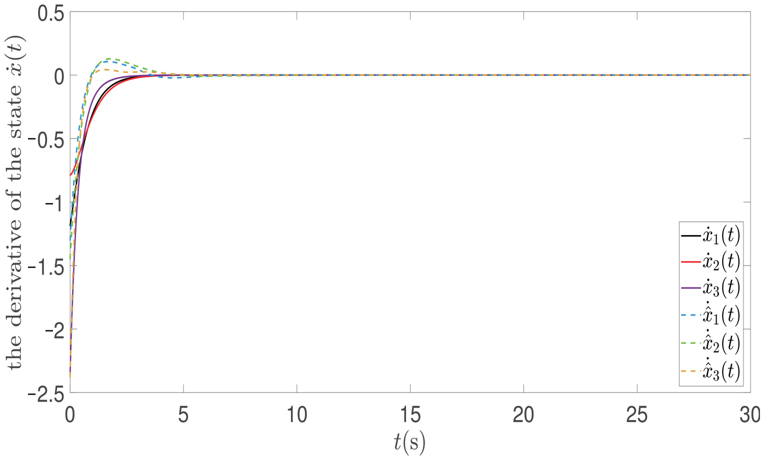

Figure 15. The derivative curves of the states and estimated states.

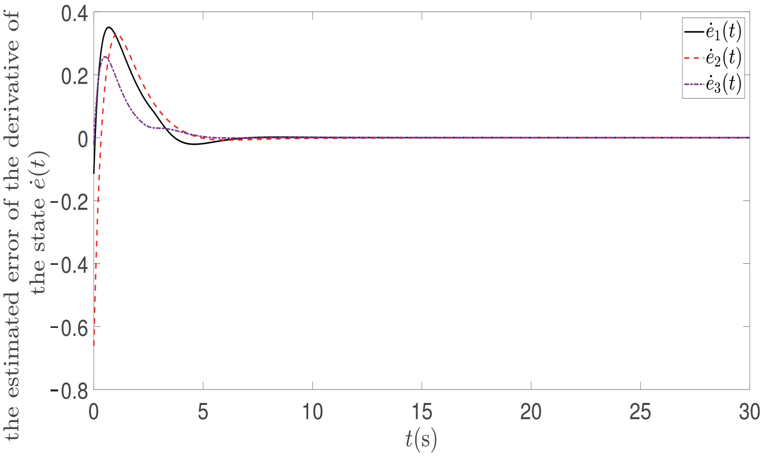

Figure 16. The error curves of the derivative of the states.

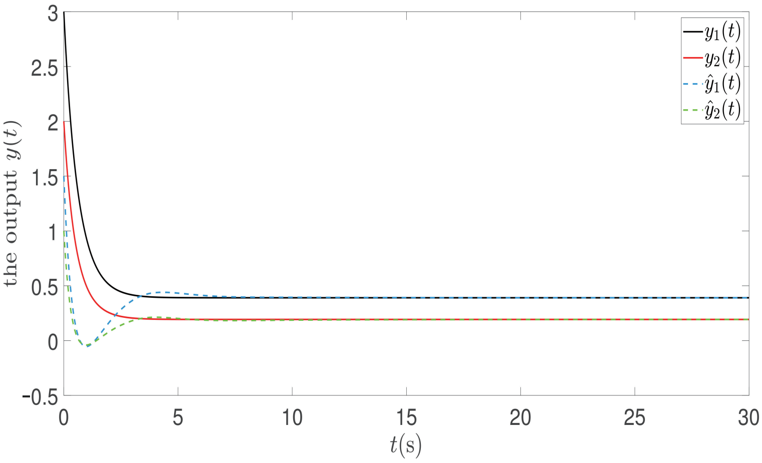

Figure 17. The output and estimated output curves.

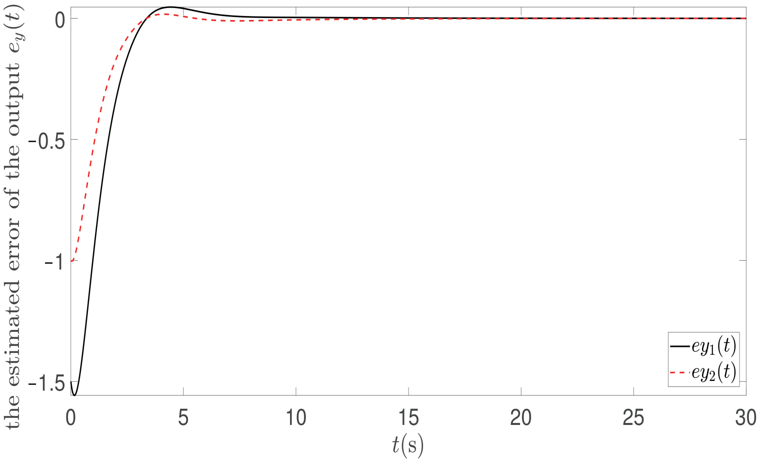

Figure 18. The error curves of the output.

In order to verify the accuracy of the estimated structure, a test system is designed based on the gain observation matrix **L**. Under the same simulation conditions as above, the effects of the adjusted weights and the random weights on the system are compared. In Figure 19, the state trajectories of the real system and the estimated system with the adjusted weights and the system with the random weights are given. Figure 20 shows the real output curve and the estimated output with the adjusted weights and the output curve with the random weights.

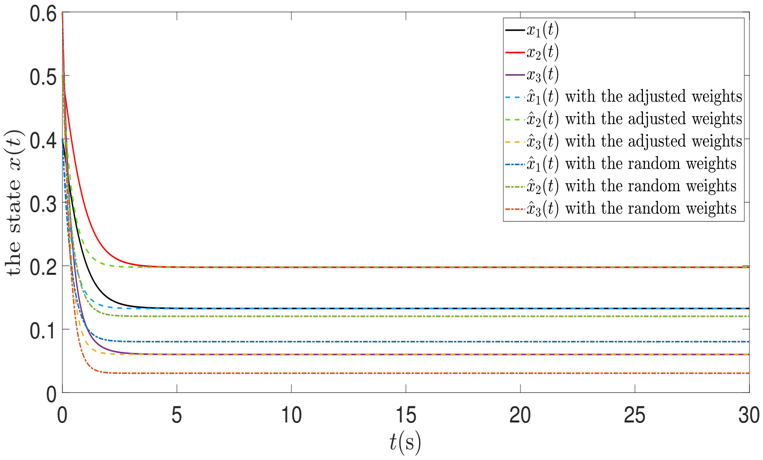

Figure 19. The state and estimated state with the adjusted weights and state with the random weights curves.

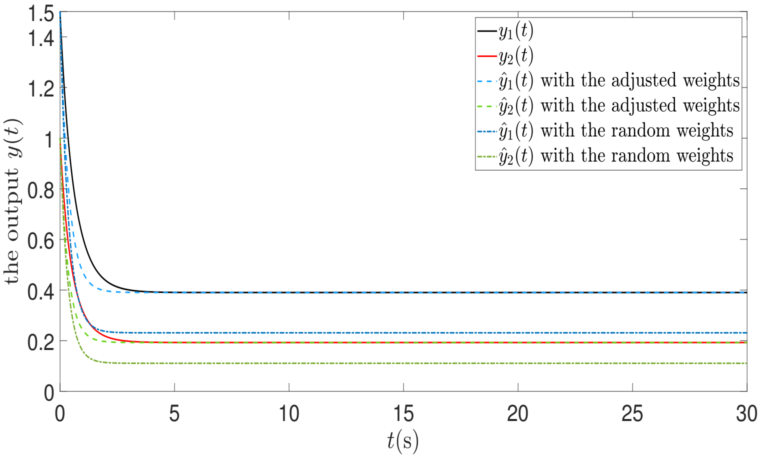

Figure 20. The output and estimated output with the adjusted weights and output with the random weights curves.

4.2. Description of Simulation Results

Figures 5 and 13 show that the estimated state vector is a good approximation of the real state vector. On the other hand, Figures 6 and 14 verify that the estimation error vector of states is going to zero. Figures 7 and 15 show that the derivative vector of estimated states is a good approximation of the derivative vector of real states. On the other hand, Figures 8 and 16 verify that the estimation error vector of derivative of states is going to zero. Figures 9 and 17 show that the estimated output vector is a good approximation of the real output vector. On the other hand, Figures 10 and 18 verify that the estimation error vector of outputs is going to zero. Figures 11 and 19 show that the estimated state vector with adaptive weights is better than that with random weights. Figures 12 and 20 show that the output vector with adaptive weights is better than that with random weights. Simulation results indicate that the state observer proposed in this paper has stronger adaptability and more accurate estimation results for memristor neural networks with model uncertainties.

5. Conclusions

The state estimation of memristor neural networks with model uncertainties is discussed in this paper. In particular, model uncertainties include time-varying delays, floating parameters and unknown functions. An improved approach based on LSTMs is proposed to deal with model uncertainties. This paper proves that the improved neural networks can approximate any nonlinear function with any error. On this basis, a full-order state observer is proposed to achieve the reconstruction of states based on the measurement output of the system. The adaptive updating laws of the weights of improved neural networks are proposed based on a LKF. By using LKF and LMIs tools, this paper obtains the asymptotic stability conditions for the error systems. The simulation results show that by using the full-order state observer and the adaptive updating laws of the weights, an accurate estimate of the solution can be obtained. The test results show that the model uncertainties can be approximated accurately. As mentioned in the introduction, the improved LSTMs designed in this paper can also be realized by crossbar array of memristor, which will be our next work.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, L.M.; visualization, supervision, M.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during and analysed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declared no potential conflict of interest with respect to the research, authorship, and/or publication of this article.

Appendix A. The Proof of Theorem 1

Proof. Considering the Equation (5), the improved neural networks are derived by using the recursive method.

Take $j = 1$, we have

$$\begin{aligned} c_1^1 &= \bar{\mathbf{W}}_{1,1}\mathbf{x}_1, \qquad h_1^1 = tanh\big(\bar{\mathbf{W}}_{1,1}\mathbf{x}_1 + \bar{b}_{1,1}\big), \\ c_1^2 &= \bar{\mathbf{W}}_{1,2}\mathbf{x}_1, \qquad h_1^2 = tanh\big(\bar{\mathbf{W}}_{1,2}\mathbf{x}_1 + \bar{b}_{1,2}\big), \\ &\vdots \\ c_1^p &= \bar{\mathbf{W}}_{1,p}\mathbf{x}_1, \qquad h_1^p = tanh\big(\bar{\mathbf{W}}_{1,p}\mathbf{x}_1 + \bar{b}_{1,p}\big). \end{aligned}$$

Then,

$$\mathbf{c}_1 = \bar{\mathbf{W}}_1\mathbf{x}_1, \qquad \mathbf{h}_1 = tanh\big(\bar{\mathbf{W}}_1\mathbf{x}_1 + \bar{\mathbf{b}}_1\big),$$

where $\mathbf{c}_1 = \begin{bmatrix} c_1^1 \\ c_1^2 \\ \vdots \\ c_1^p \end{bmatrix}$, $\bar{\mathbf{W}}_1 = \begin{bmatrix} \bar{\mathbf{W}}_{1,1} \\ \bar{\mathbf{W}}_{1,2} \\ \vdots \\ \bar{\mathbf{W}}_{1,p} \end{bmatrix} = \begin{bmatrix} \bar{W}_{1,1,1} & \bar{W}_{1,1,2} & \cdots & \bar{W}_{1,1,n} \\ \bar{W}_{1,2,1} & \bar{W}_{1,2,2} & \cdots & \bar{W}_{1,2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{W}_{1,p,1} & \bar{W}_{1,p,2} & \cdots & \bar{W}_{1,p,n} \end{bmatrix}$, $\mathbf{h}_1 = \begin{bmatrix} h_1^1 \\ h_1^2 \\ \vdots \\ h_1^p \end{bmatrix}$, $\bar{\mathbf{b}}_1 = \begin{bmatrix} \bar{b}_{1,1} \\ \bar{b}_{1,2} \\ \vdots \\ \bar{b}_{1,p} \end{bmatrix}$.

And we have

$$
\begin{aligned}
&y_{1,1} = \bar{V}_{1,1,1} \cdot tanh\big(\bar{\mathbf{W}}_{1,1}\mathbf{x}_1 + \bar{b}_{1,1}\big) + \bar{V}_{1,2,1} \cdot tanh\big(\bar{\mathbf{W}}_{1,2}\mathbf{x}_1 + \bar{b}_{1,2}\big) + \cdots \\
&+ \bar{V}_{1,p,1} \cdot tanh\big(\bar{\mathbf{W}}_{1,p}\mathbf{x}_1 + \bar{b}_{1,p}\big), \\
&y_{1,2} = \bar{V}_{1,1,2} \cdot tanh\big(\bar{\mathbf{W}}_{1,1}\mathbf{x}_1 + \bar{b}_{1,1}\big) + \bar{V}_{1,2,2} \cdot tanh\big(\bar{\mathbf{W}}_{1,2}\mathbf{x}_1 + \bar{b}_{1,2}\big) + \cdots \\
&+ \bar{V}_{1,p,2} \cdot tanh\big(\bar{\mathbf{W}}_{1,p}\mathbf{x}_1 + \bar{b}_{1,p}\big), \\
&\vdots \\
&y_{1,m} = \bar{V}_{1,1,m} \cdot tanh\big(\bar{\mathbf{W}}_{1,1}\mathbf{x}_1 + \bar{b}_{1,1}\big) + \bar{V}_{1,2,m} \cdot tanh\big(\bar{\mathbf{W}}_{1,2}\mathbf{x}_1 + \bar{b}_{1,2}\big) + \cdots \\
&+ \bar{V}_{1,p,m} \cdot tanh\big(\bar{\mathbf{W}}_{1,p}\mathbf{x}_1 + \bar{b}_{1,p}\big).
\end{aligned}
$$

Then,

$$\mathbf{y}_1 = \bar{\mathbf{V}}_1^{\mathrm{T}} \cdot tanh\big(\bar{\mathbf{W}}_1\mathbf{x}_1 + \bar{\mathbf{b}}_1\big),$$

where $\mathbf{y}_1 = \begin{bmatrix} y_{1,1} \\ y_{1,2} \\ \vdots \\ y_{1,m} \end{bmatrix}$, $\bar{\mathbf{V}}_1 = \begin{bmatrix} \bar{V}_{1,1,1} & \bar{V}_{1,1,2} & \cdots & \bar{V}_{1,1,m} \\ \bar{V}_{1,2,1} & \bar{V}_{1,2,2} & \cdots & \bar{V}_{1,2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{V}_{1,p,1} & \bar{V}_{1,p,2} & \cdots & \bar{V}_{1,p,m} \end{bmatrix}$.

Take $j = 2$, we have

$$
\begin{aligned}
&c_2^1 = \bar{\mathbf{W}}_{2,1}\mathbf{x}_2 + c_1^1 \cdot \sigma(\mathbf{x}_2) = \bar{\mathbf{W}}_{2,1}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,1}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2), \\
&h_2^1 = tanh\big(\bar{\mathbf{W}}_{2,1}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,1}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,1}\big), \\
&c_2^2 = \bar{\mathbf{W}}_{2,2}\mathbf{x}_2 + c_1^2 \cdot \sigma(\mathbf{x}_2) = \bar{\mathbf{W}}_{2,2}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,2}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2), \\
&h_2^2 = tanh\big(\bar{\mathbf{W}}_{2,2}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,2}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,2}\big), \\
&\vdots \\
&c_2^p = \bar{\mathbf{W}}_{2,p}\mathbf{x}_2 + c_1^p \cdot \sigma(\mathbf{x}_2) = \bar{\mathbf{W}}_{2,p}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,p}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2), \\
&h_2^p = tanh\big(\bar{\mathbf{W}}_{2,p}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,p}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,p}\big).
\end{aligned}
$$

Then,

$$\mathbf{c}_2 = \bar{\mathbf{W}}_2\mathbf{x}_2 + \bar{\mathbf{W}}_1\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2), \qquad \mathbf{h}_2 = tanh\big(\bar{\mathbf{W}}_2\mathbf{x}_2 + \bar{\mathbf{W}}_1\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{\mathbf{b}}_2\big),$$

where $\mathbf{c}_2 = \begin{bmatrix} c_2^1 \\ c_2^2 \\ \vdots \\ c_2^p \end{bmatrix}$, $\bar{\mathbf{W}}_2 = \begin{bmatrix} \bar{\mathbf{W}}_{2,1} \\ \bar{\mathbf{W}}_{2,2} \\ \vdots \\ \bar{\mathbf{W}}_{2,p} \end{bmatrix} = \begin{bmatrix} \bar{W}_{2,1,1} & \bar{W}_{2,1,2} & \cdots & \bar{W}_{2,1,n} \\ \bar{W}_{2,2,1} & \bar{W}_{2,2,2} & \cdots & \bar{W}_{2,2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{W}_{2,p,1} & \bar{W}_{2,p,2} & \cdots & \bar{W}_{2,p,n} \end{bmatrix}$, $\mathbf{h}_2 = \begin{bmatrix} h_2^1 \\ h_2^2 \\ \vdots \\ h_2^p \end{bmatrix}$,
$\bar{\mathbf{b}}_2 = \begin{bmatrix} \bar{b}_{2,1} \\ \bar{b}_{2,2} \\ \vdots \\ \bar{b}_{2,p} \end{bmatrix}$.

And we have

$$\begin{aligned}
y_{2,1} &= \bar{V}_{2,1,1} \cdot tanh\big(\bar{\mathbf{W}}_{2,1}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,1}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,1}\big) + \bar{V}_{2,2,1} \cdot tanh(\bar{\mathbf{W}}_{2,2}\mathbf{x}_2 \\
&+ \bar{\mathbf{W}}_{1,2}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,2}) + \cdots + \bar{V}_{2,p,1} \cdot tanh\big(\bar{\mathbf{W}}_{2,p}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,p}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,p}\big), \\
y_{2,2} &= \bar{V}_{2,1,2} \cdot tanh\big(\bar{\mathbf{W}}_{2,1}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,1}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,1}\big) + \bar{V}_{2,2,2} \cdot tanh(\bar{\mathbf{W}}_{2,2}\mathbf{x}_2 \\
&+ \bar{\mathbf{W}}_{1,2}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,2}) + \cdots + \bar{V}_{2,p,2} \cdot tanh\big(\bar{\mathbf{W}}_{2,p}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,p}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,p}\big), \\
&\vdots \\
y_{2,m} &= \bar{V}_{2,1,m} \cdot tanh\big(\bar{\mathbf{W}}_{2,1}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,1}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,1}\big) + \bar{V}_{2,2,m} \cdot tanh(\bar{\mathbf{W}}_{2,2}\mathbf{x}_2 \\
&+ \bar{\mathbf{W}}_{1,2}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,2}) + \cdots + \bar{V}_{2,p,m} \cdot \cdot tanh\big(\bar{\mathbf{W}}_{2,p}\mathbf{x}_2 + \bar{\mathbf{W}}_{1,p}\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{b}_{2,p}\big).
\end{aligned}$$

Then,

$$\mathbf{y}_2 = \bar{\mathbf{V}}_2^{\mathrm{T}} \cdot tanh\big(\bar{\mathbf{W}}_2\mathbf{x}_2 + \bar{\mathbf{W}}_1\mathbf{x}_1 \cdot \sigma(\mathbf{x}_2) + \bar{\mathbf{b}}_2\big),$$

where $\mathbf{y}_2 = \begin{bmatrix} y_{2,1} \\ y_{2,2} \\ \vdots \\ y_{2,m} \end{bmatrix}$, $\bar{\mathbf{V}}_2 = \begin{bmatrix} \bar{V}_{2,1,1} & \bar{V}_{2,1,2} & \cdots & \bar{V}_{2,1,m} \\ \bar{V}_{2,2,1} & \bar{V}_{2,2,2} & \cdots & \bar{V}_{2,2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{V}_{2,p,1} & \bar{V}_{2,p,2} & \cdots & \bar{V}_{2,p,m} \end{bmatrix}$.

Take $j = t$, we have

$$\begin{aligned}
c_t^1 &= \bar{\mathbf{W}}_{t,1}\mathbf{x}_t + c_{t-1}^1 \cdot \sigma(\mathbf{x}_t) \\
&= \bar{\mathbf{W}}_{t,1}\mathbf{x}_t + \Big[\bar{\mathbf{W}}_{t-1,1}\mathbf{x}_{t-1} + c_{t-2}^1 \cdot \sigma(\mathbf{x}_{t-1})\Big] \cdot \sigma(\mathbf{x}_t) \\
&= \bar{\mathbf{W}}_{t,1}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,1}\mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j)\right], \\
h_t^1 &= tanh\left\{\bar{\mathbf{W}}_{t,1}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,1}\mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j)\right] + \bar{b}_{t,1}\right\}, \\
c_t^2 &= \bar{\mathbf{W}}_{t,2}\mathbf{x}_t + c_{t-1}^2 \cdot \sigma(\mathbf{x}_t) \\
&= \bar{\mathbf{W}}_{t,2}\mathbf{x}_t + \Big[\bar{\mathbf{W}}_{t-1,2}\mathbf{x}_{t-1} + c_{t-2}^2 \cdot \sigma(\mathbf{x}_{t-1})\Big] \cdot \sigma(\mathbf{x}_t) \\
&= \bar{\mathbf{W}}_{t,2}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,2}\mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j)\right], \\
h_t^2 &= tanh\left\{\bar{\mathbf{W}}_{t,2}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,2}\mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j)\right] + \bar{b}_{t,2}\right\}, \\
&\vdots \\
c_t^p &= \bar{\mathbf{W}}_{t,p}\mathbf{x}_t + c_{t-1}^p \cdot \sigma(\mathbf{x}_t) \\
&= \bar{\mathbf{W}}_{t,p}\mathbf{x}_t + \Big[\bar{\mathbf{W}}_{t-1,p}\mathbf{x}_{t-1} + c_{t-2}^p \cdot \sigma(\mathbf{x}_{t-1})\Big] \cdot \sigma(\mathbf{x}_t) \\
&= \bar{\mathbf{W}}_{t,p}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,p}\mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j)\right], \\
h_t^p &= tanh\left\{\bar{\mathbf{W}}_{t,p}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,p}\mathbf{x}_i \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j)\right] + \bar{b}_{t,p}\right\}.
\end{aligned}$$

Then,

$$\mathbf{c}_t = \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right],$$

$$\mathbf{h}_t = tanh\left\{\bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t\right\},$$

where $\mathbf{c}_t = \begin{bmatrix} c_t^1 \\ c_t^2 \\ \vdots \\ c_t^p \end{bmatrix}$, $\bar{\mathbf{W}}_t = \begin{bmatrix} \bar{\mathbf{W}}_{t,1} \\ \bar{\mathbf{W}}_{t,2} \\ \vdots \\ \bar{\mathbf{W}}_{t,p} \end{bmatrix} = \begin{bmatrix} \bar{W}_{t,1,1} & \bar{W}_{t,1,2} & \cdots & \bar{W}_{t,1,n} \\ \bar{W}_{t,2,1} & \bar{W}_{t,2,2} & \cdots & \bar{W}_{t,2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{W}_{t,p,1} & \bar{W}_{t,p,2} & \cdots & \bar{W}_{t,p,n} \end{bmatrix}$, $\mathbf{h}_t = \begin{bmatrix} h_t^1 \\ h_t^2 \\ \vdots \\ h_t^p \end{bmatrix}$, $\bar{\mathbf{b}}_t = \begin{bmatrix} \bar{b}_{t,1} \\ \bar{b}_{t,2} \\ \vdots \\ \bar{b}_{t,p} \end{bmatrix}$.

And we have

$$\begin{aligned}
y_{t,1} = {} & \bar{V}_{t,1,1} \cdot tanh\left\{\bar{\mathbf{W}}_{t,1}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,1}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,1}\right\} \\
& + \bar{V}_{t,2,1} \cdot tanh\left\{\bar{\mathbf{W}}_{t,2}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,2}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,2}\right\} + \cdots \\
& + \bar{V}_{t,p,1} \cdot tanh\left\{\bar{\mathbf{W}}_{t,p}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,p}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,p}\right\}, \\
y_{t,2} = {} & \bar{V}_{t,1,2} \cdot tanh\left\{\bar{\mathbf{W}}_{t,1}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,1}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,1}\right\} \\
& + \bar{V}_{t,2,2} \cdot tanh\left\{\bar{\mathbf{W}}_{t,2}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,2}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,2}\right\} + \cdots \\
& + \bar{V}_{t,p,2} \cdot tanh\left\{\bar{\mathbf{W}}_{t,p}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,p}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,p}\right\}, \\
& \vdots \\
y_{t,m} = {} & \bar{V}_{t,1,m} \cdot tanh\left\{\bar{\mathbf{W}}_{t,1}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,1}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,1}\right\} \\
& + \bar{V}_{t,2,m} \cdot tanh\left\{\bar{\mathbf{W}}_{t,2}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,2}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,2}\right\} + \cdots \\
& + \bar{V}_{t,p,m} \cdot tanh\left\{\bar{\mathbf{W}}_{t,p}\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_{i,p}\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{b}_{t,p}\right\}.
\end{aligned}$$

Then,

$$\mathbf{y}_t = \bar{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{\bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t\right\},$$

where $\mathbf{y}_t = \begin{bmatrix} y_{t,1} \\ y_{t,2} \\ \vdots \\ y_{t,m} \end{bmatrix}$, $\bar{\mathbf{V}}_t = \begin{bmatrix} \bar{V}_{t,1,1} & \bar{V}_{t,1,2} & \cdots & \bar{V}_{t,1,m} \\ \bar{V}_{t,2,1} & \bar{V}_{t,2,2} & \cdots & \bar{V}_{t,2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{V}_{t,p,1} & \bar{V}_{t,p,2} & \cdots & \bar{V}_{t,p,m} \end{bmatrix}$.

Take $\sigma(\mathbf{x}_j) = 1, (j = 1, 2, \cdots, t)$, and $\mathbf{y}_t$ can be reconstituted as follows,

$$\mathbf{y}_t = \bar{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\{\bar{\mathbf{W}}\mathbf{x} + \bar{\mathbf{b}}_t\},$$

where $\bar{\mathbf{W}} = [\bar{\mathbf{W}}_t, \bar{\mathbf{W}}_{t-1}, \cdots, \bar{\mathbf{W}}_1]$, $\mathbf{x} = [\mathbf{x}_t, \mathbf{x}_{t-1}, \cdots, \mathbf{x}_1]^{\mathrm{T}}$.

It can be found that the formula is the basic form of multiple neural networks. Therefore, the improved LSTMs can be converted into multiple neural networks. Considering Lemma 1, multiple neural networks can approximate any nonlinear function with any error. Considering the improved LSTMs, since it can be converted into the multiple neural networks, the lemma satisfied by the multiple neural networks can also be satisfied by the improved LSTMs. Therefore, Theorem 1 can be obtained according to the form of Lemma 1. □

Appendix B. The Proof of Theorem 2

Proof. Take a Lyapunov-Krasovskii function,

$$V(t, \mathbf{e}(t)) = \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\mathbf{e}(t) + \int_{t-\tau(t)}^{t} \mathbf{e}^{\mathrm{T}}(s)\mathbf{Q}\mathbf{e}(s),$$

where $\mathbf{P}$ and $\mathbf{Q}$ are positive definite matrices.

Based on the error system (9), the derivative of $V(t, \mathbf{e}(t))$ can be obtained as follows,

$$\begin{aligned}
\dot{V}(t, \mathbf{e}(t)) &= \dot{\mathbf{e}}^{\mathrm{T}}(t)\mathbf{P}\mathbf{e}(t) + \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\dot{\mathbf{e}}(t) + \mathbf{e}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{e}(t) - \mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{Q}\mathbf{e}(t-\tau(t)) \\
&= [-(\mathbf{A} + \mathbf{L}\mathbf{H})\mathbf{e}(t) + (\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t))))]^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
&+ \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}[-(\mathbf{A} + \mathbf{L}\mathbf{H})\mathbf{e}(t) + (\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t))))] \\
&+ \mathbf{e}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{e}(t) - \mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{Q}\mathbf{e}(t-\tau(t)) \\
&= -\mathbf{e}^{\mathrm{T}}(t)(\mathbf{A} + \mathbf{L}\mathbf{H})^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) - \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}(\mathbf{A} + \mathbf{L}\mathbf{H})\mathbf{e}(t) + \mathbf{e}^{\mathrm{T}}(t)\mathbf{Q}\mathbf{e}(t) \\
&+ \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}(\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))) - \mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{Q}\mathbf{e}(t-\tau(t)) \\
&+ (\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t))))^{\mathrm{T}}\mathbf{P}\mathbf{e}(t).
\end{aligned}$$

By using Assumption 2 and Lemma 2, for a positive definite diagonal matrix $\mathbf{F}$, we have

$$\begin{aligned}
&[\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))]^{\mathrm{T}}\mathbf{F}[\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))] \\
&= \sum_{i=1}^{n}\Big\{F_i[\bar{K}_i - K_i(\mathbf{x}(t), \mathbf{x}(t-\tau(t)))]^2\Big\} \\
&\leqslant \sum_{i=1}^{n}\Big\{F_i\Big[\mathbf{K}_{L1}^{\mathrm{T}}|\mathbf{x}(t) - \hat{\mathbf{x}}(t)| + \mathbf{K}_{L2}^{\mathrm{T}}|\mathbf{x}(t-\tau(t)) - \hat{\mathbf{x}}(t-\tau(t))|\Big]^2\Big\} \\
&\leqslant \sum_{i=1}^{n}\Big\{F_i\Big[\mathbf{e}^{\mathrm{T}}(t)\mathbf{K}_{L1}\mathbf{K}_{L1}^{\mathrm{T}}\mathbf{e}(t) + \mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{K}_{L2}\mathbf{K}_{L2}^{\mathrm{T}}\mathbf{e}(t-\tau(t))\Big]\Big\} \\
&+ \sum_{i=1}^{n}\Big\{F_i\Big[2|\mathbf{e}^{\mathrm{T}}(t)|\mathbf{K}_{L1}\mathbf{K}_{L2}^{\mathrm{T}}|\mathbf{e}(t-\tau(t))|\Big]\Big\} \\
&\leqslant 2\mathbf{e}^{\mathrm{T}}(t)\mathbf{K}_{L1}tr\{\mathbf{F}\}\mathbf{K}_{L1}^{\mathrm{T}}\mathbf{e}(t) + 2\mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{K}_{L2}tr\{\mathbf{F}\}\mathbf{K}_{L2}^{\mathrm{T}}\mathbf{e}(t-\tau(t)).
\end{aligned}$$

And there exists a positive definite matrix $\mathbf{M}$ to satisfy the following equation

$$2\bigg[\int_{t-\tau(t)}^{t} \dot{\mathbf{e}}(s)ds - \mathbf{e}(t) + \mathbf{e}(t-\tau(t))\bigg]\mathbf{M}\bigg[-\int_{t-\tau(t)}^{t} \dot{\mathbf{e}}^{\mathrm{T}}(s)ds - \mathbf{e}^{\mathrm{T}}(t-\tau(t))\bigg] = 0.$$

For a real constant $\delta > 0$, combining with the above formulas, we have

$$\begin{aligned}
\dot{V}(t,\mathbf{e}(t)) \leqslant\ & -\mathbf{e}^{\mathrm{T}}(t)(\mathbf{A}+\mathbf{LH})^{\mathrm{T}}\mathbf{Pe}(t) - \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}(\mathbf{A}+\mathbf{LH})\mathbf{e}(t) + \mathbf{e}^{\mathrm{T}}(t)\mathbf{Qe}(t) \\
& + \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}(\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))) - \mathbf{e}^{\mathrm{T}}(t-\tau(t))(1-\delta)\mathbf{Qe}(t-\tau(t)) \\
& + (\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t))))^{\mathrm{T}}\mathbf{Pe}(t) + 2\mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{K}_{L2}tr\{\mathbf{F}\}\mathbf{K}_{L2}^{\mathrm{T}}\mathbf{e}(t-\tau(t)) \\
& + 2\mathbf{e}^{\mathrm{T}}(t)\mathbf{K}_{L1}tr\{\mathbf{F}\}\mathbf{K}_{L1}^{\mathrm{T}}\mathbf{e}(t) - [\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))]^{\mathrm{T}}\mathbf{F}[\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))] \\
& + 2\left[\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}(s)ds - \mathbf{e}(t) + \mathbf{e}(t-\tau(t))\right]\mathbf{M}\left[-\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}^{\mathrm{T}}(s)ds - \mathbf{e}^{\mathrm{T}}(t-\tau(t))\right] \\
\leqslant\ & \mathbf{e}^{\mathrm{T}}(t)\left[-\mathbf{A}^{\mathrm{T}}\mathbf{P} - \mathbf{H}^{\mathrm{T}}\mathbf{L}^{\mathrm{T}}\mathbf{P} - \mathbf{PA} - \mathbf{PLH} + \mathbf{Q} + 2\mathbf{K}_{L1}tr\{\mathbf{F}\}\mathbf{K}_{L1}^{\mathrm{T}}\right]\mathbf{e}(t) \\
& + \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}(\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))) + (\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t))))^{\mathrm{T}}\mathbf{Pe}(t) \\
& - [\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))]^{\mathrm{T}}\mathbf{F}[\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))] \\
& + \mathbf{e}^{\mathrm{T}}(t-\tau(t))\left[2\mathbf{K}_{L2}tr\{\mathbf{F}\}\mathbf{K}_{L2}^{\mathrm{T}} - (1-\delta)\mathbf{Q} - 2\mathbf{M}\right]\mathbf{e}(t-\tau(t)) \\
& + 2\mathbf{e}^{\mathrm{T}}(t-\tau(t)\mathbf{Me}(t) + 2\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}^{\mathrm{T}}(s)ds\mathbf{Me}(t) - 2\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}^{\mathrm{T}}(s)ds\mathbf{Me}(t-\tau(t)) \\
& - 2\mathbf{e}^{\mathrm{T}}(t-\tau(t))\mathbf{M}\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}(s)ds - 2\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}^{\mathrm{T}}(s)ds\mathbf{M}\int_{t-\tau(t)}^{t}\dot{\mathbf{e}}(s)ds \\
\leqslant\ & \boldsymbol{\jmath}^{\mathrm{T}} \cdot \begin{bmatrix} \mathbf{\Omega}_1 & \mathbf{P} & \mathbf{M} & \mathbf{M} \\ \mathbf{P} & -\mathbf{F} & 0 & 0 \\ \mathbf{M} & 0 & \mathbf{\Omega}_2 & -2\mathbf{M} \\ \mathbf{M} & 0 & -2\mathbf{M} & -2\mathbf{M} \end{bmatrix} \cdot \boldsymbol{\jmath},
\end{aligned}$$

where $\boldsymbol{\jmath} = \left[\mathbf{e}(t), (\bar{\mathbf{K}} - \mathbf{K}(\mathbf{x}(t),\mathbf{x}(t-\tau(t)))), \mathbf{e}(t-\tau(t)), \int_{t-\tau(t)}^{t}\dot{\mathbf{e}}(s)ds\right]^{\mathrm{T}}$, $\mathbf{\Omega}_1 = -\mathbf{A}^{\mathrm{T}}\mathbf{P} - \mathbf{H}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}} - \mathbf{PA} - \mathbf{GH} + \mathbf{Q} + 2\mathbf{K}_{L1}tr\{\mathbf{F}\}\mathbf{K}_{L1}^{\mathrm{T}}$, $\mathbf{\Omega}_2 = 2\mathbf{K}_{L2}tr\{\mathbf{F}\}\mathbf{K}_{L2}^{\mathrm{T}} - (1-\delta)\mathbf{Q} - 2\mathbf{M}$, and $\mathbf{G} = \mathbf{PL}$.

Considering the above inequality, take

$$\begin{bmatrix} \mathbf{\Omega}_1 & \mathbf{P} & \mathbf{M} & \mathbf{M} \\ \mathbf{P} & -\mathbf{F} & 0 & 0 \\ \mathbf{M} & 0 & \mathbf{\Omega}_2 & -2\mathbf{M} \\ \mathbf{M} & 0 & -2\mathbf{M} & -2\mathbf{M} \end{bmatrix} < 0,$$

then the error system (9) is asymptotically stable, and the proof of Theorem 2 is completed. □

Appendix C. The Proof of Theorem 3

Proof. Based on the Equation (15), take a Lyapunov–Krasovskii function(LKF),

$$\begin{aligned}
V(t,\mathbf{e}(t)) &= \mathbf{e}^{\mathrm{T}}(t)\mathbf{Pe}(t) + tr\{\tilde{\mathbf{V}}_t^{\mathrm{T}} \cdot \tilde{\mathbf{V}}_t\} + tr\{\tilde{\mathbf{W}}_t^{\mathrm{T}} \cdot \tilde{\mathbf{W}}_t\} + \tilde{\mathbf{b}}_t^{\mathrm{T}} \cdot \tilde{\mathbf{b}}_t \\
&\quad + \sum_{i=1}^{t-1}\left[tr\{\tilde{\mathbf{W}}_i^{\mathrm{T}} \cdot \tilde{\mathbf{W}}_i\} \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right].
\end{aligned}$$

By using the error system (14) and the Equation (15), the derivative of V (t; e(t)) can be obtained as follows,

$$\begin{aligned}
\dot{V}(t,\mathbf{e}(t)) &= \dot{\mathbf{e}}^{\mathrm{T}}(t)\mathbf{P}\mathbf{e}(t) + \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\dot{\mathbf{e}}(t) + tr\{\tilde{\mathbf{V}}_t^{\mathrm{T}} \cdot \dot{\hat{\mathbf{V}}}_t\} + tr\{\tilde{\mathbf{W}}_t^{\mathrm{T}} \cdot \dot{\hat{\mathbf{W}}}_t\} + \tilde{\mathbf{b}}_t^{\mathrm{T}} \cdot \dot{\hat{\mathbf{b}}}_t \\
&\quad + \sum_{i=1}^{t-1}\left[tr\{\tilde{\mathbf{W}}_i^{\mathrm{T}} \cdot \dot{\hat{\mathbf{W}}}_i\} \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j) \right] \\
&= \mathbf{e}^{\mathrm{T}}(t)\Big[-(\mathbf{A}+\mathbf{LH})^{\mathrm{T}}\mathbf{P} - \mathbf{P}(\mathbf{A}+\mathbf{LH})\Big]\mathbf{e}(t) + \big(\hat{\mathbf{K}} - \bar{\mathbf{K}}\big)^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) - \epsilon_1^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
&\quad + \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\big(\hat{\mathbf{K}} - \bar{\mathbf{K}}\big) - \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\epsilon_1 + tr\{\tilde{\mathbf{V}}_t^{\mathrm{T}} \cdot \dot{\hat{\mathbf{V}}}_t\} + tr\{\tilde{\mathbf{W}}_t^{\mathrm{T}} \cdot \dot{\hat{\mathbf{W}}}_t\} + \tilde{\mathbf{b}}_t^{\mathrm{T}} \cdot \dot{\hat{\mathbf{b}}}_t \\
&\quad + \sum_{i=1}^{t-1}\left[tr\{\tilde{\mathbf{W}}_i^{\mathrm{T}} \cdot \dot{\hat{\mathbf{W}}}_i\} \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j) \right].
\end{aligned}$$

Combining (8) and (13), it can be obtained

$$\begin{aligned}
\hat{\mathbf{K}} - \bar{\mathbf{K}} &= \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \hat{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\hat{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \hat{\mathbf{b}}_t \right\} \\
&\quad - \bar{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t \right\} \\
&= \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \hat{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\hat{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \hat{\mathbf{b}}_t \right\} \\
&\quad - \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \hat{\mathbf{b}}_t \right\} \\
&\quad + \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \hat{\mathbf{b}}_t \right\} \\
&\quad - \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t \right\} \\
&\quad + \hat{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t \right\} \\
&\quad - \bar{\mathbf{V}}_t^{\mathrm{T}} \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t \right\} \\
&\leqslant 2\hat{\mathbf{V}}_t^{\mathrm{T}} \cdot \left\{ (\hat{\mathbf{W}}_t - \bar{\mathbf{W}}_t)\mathbf{x}_t + \sum_{i=1}^{t-1}\left[(\hat{\mathbf{W}}_i - \bar{\mathbf{W}}_i)\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] \right\} + 2\hat{\mathbf{V}}_t^{\mathrm{T}} \cdot (\hat{\mathbf{b}}_t - \bar{\mathbf{b}}_t) \\
&\quad + (\hat{\mathbf{V}}_t^{\mathrm{T}} - \bar{\mathbf{V}}_t^{\mathrm{T}}) \cdot tanh\left\{ \bar{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\bar{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \bar{\mathbf{b}}_t \right\} \\
&\leqslant 2\hat{\mathbf{V}}_t^{\mathrm{T}} \cdot \left\{ \tilde{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\tilde{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] \right\} + 2\hat{\mathbf{V}}_t^{\mathrm{T}} \cdot \tilde{\mathbf{b}}_t + \tilde{\mathbf{V}}_t^{\mathrm{T}} \cdot \hat{\mathbf{S}}_t - \tilde{\mathbf{V}}_t^{\mathrm{T}} \cdot \epsilon_2,
\end{aligned}$$

where $\hat{\mathbf{S}}_t = tanh\left\{ \hat{\mathbf{W}}_t\mathbf{x}_t + \sum_{i=1}^{t-1}\left[\hat{\mathbf{W}}_i\mathbf{x}_i \cdot \prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] + \hat{\mathbf{b}}_t \right\}$ and $\epsilon_2 = \hat{\mathbf{S}}_t - \bar{\mathbf{S}}_t$.

Considering the above equations, we have

$$
\begin{aligned}
\dot{V}(t,\mathbf{e}(t)) \leqslant\ & \mathbf{e}^{\mathrm{T}}(t)\Big[-(\mathbf{A}+\mathbf{LH})^{\mathrm{T}}\mathbf{P}-\mathbf{P}(\mathbf{A}+\mathbf{LH})\Big]\mathbf{e}(t)-\mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\epsilon_1-\epsilon_1^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
& +4\Big[\hat{\mathbf{V}}_t^{\mathrm{T}}\tilde{\mathbf{W}}_t\mathbf{x}_t\Big]^{\mathrm{T}}\mathbf{P}\mathbf{e}(t)+4\sum_{i=1}^{t-1}\left[(\hat{\mathbf{V}}_t^{\mathrm{T}}\tilde{\mathbf{W}}_i\mathbf{x}_i)^{\mathrm{T}}\mathbf{P}\mathbf{e}(t)\cdot\prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] \\
& +4\Big[\hat{\mathbf{V}}_t^{\mathrm{T}}\cdot\tilde{\mathbf{b}}_t\Big]^{\mathrm{T}}\mathbf{P}\mathbf{e}(t)+2\Big[\tilde{\mathbf{V}}_t^{\mathrm{T}}\cdot\hat{\mathbf{S}}_t\Big]^{\mathrm{T}}\mathbf{P}\mathbf{e}(t)-2\Big[\tilde{\mathbf{V}}_t^{\mathrm{T}}\cdot\epsilon_2\Big]^{\mathrm{T}}\mathbf{P}\mathbf{e}(t)+tr\{\tilde{\mathbf{V}}_t^{\mathrm{T}}\cdot\dot{\hat{\mathbf{V}}}_t\} \\
& +tr\{\tilde{\mathbf{W}}_t^{\mathrm{T}}\cdot\dot{\hat{\mathbf{W}}}_t\}+\tilde{\mathbf{b}}_t^{\mathrm{T}}\cdot\dot{\hat{\mathbf{b}}}_t+\sum_{i=1}^{t-1}\left[tr\{\tilde{\mathbf{W}}_i^{\mathrm{T}}\cdot\dot{\hat{\mathbf{W}}}_i\}\cdot\prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right] \\
\leqslant\ & \mathbf{e}^{\mathrm{T}}(t)\Big[-(\mathbf{A}+\mathbf{LH})^{\mathrm{T}}\mathbf{P}-\mathbf{P}(\mathbf{A}+\mathbf{LH})\Big]\mathbf{e}(t)-\mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\epsilon_1-\epsilon_1^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
& +tr\Big\{\tilde{\mathbf{W}}_t^{\mathrm{T}}\cdot\Big[4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t)\mathbf{x}_t^{\mathrm{T}}+\dot{\hat{\mathbf{W}}}_t\Big]\Big\}+tr\Big\{\tilde{\mathbf{V}}_t^{\mathrm{T}}\cdot\Big[2\hat{\mathbf{S}}_t\mathbf{e}^{\mathrm{T}}(t)\mathbf{P}+\dot{\hat{\mathbf{V}}}_t\Big]\Big\} \\
& +\sum_{i=1}^{t-1}\left[tr\Big\{\tilde{\mathbf{W}}_i^{\mathrm{T}}\cdot\Big[4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t)\mathbf{x}_i^{\mathrm{T}}+\dot{\hat{\mathbf{W}}}_i\Big]\Big\}\cdot\prod_{j=i+1}^{t}\sigma(\mathbf{x}_j)\right]-2\epsilon_2^{\mathrm{T}}\tilde{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t) \\
& +\tilde{\mathbf{b}}_t^{\mathrm{T}}\cdot\Big[4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t)+\dot{\hat{\mathbf{b}}}_t\Big].
\end{aligned}
$$

Then, adaptive updating laws of the weights are given as follows

$$
\begin{aligned}
\dot{\hat{\mathbf{W}}}_i &= \mathbf{N}_{w\_i}\cdot\mathbf{e}^{\mathrm{T}}(t)-4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t)\mathbf{x}_i^{\mathrm{T}} \quad (i=1,2,\cdots,t) \\
\dot{\hat{\mathbf{V}}}_t &= \mathbf{N}_{v\_t}\cdot\mathbf{e}^{\mathrm{T}}(t)-2\hat{\mathbf{S}}_t\mathbf{e}^{\mathrm{T}}(t)\mathbf{P} \\
\dot{\hat{\mathbf{b}}}_t &= \mathbf{N}_{b\_t}\cdot\mathbf{e}(t)-4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t),
\end{aligned}
$$

where $\mathbf{N}_{w\_i}=\begin{bmatrix} N_{w\_i,1} \\ N_{w\_i,2} \\ \vdots \\ N_{w\_i,p} \end{bmatrix}$ and $\mathbf{N}_{v\_t}=\begin{bmatrix} N_{v\_t,1} \\ N_{v\_t,2} \\ \vdots \\ N_{v\_t,p} \end{bmatrix}$ denote the design vectors;
$\mathbf{N}_{b\_t}=\begin{bmatrix} N_{b\_t,1,1} & N_{b\_t,1,2} & \cdots & N_{b\_t,1,n} \\ N_{b\_t,2,1} & N_{b\_t,2,2} & \cdots & N_{b\_t,2,n} \\ \vdots & \vdots & \ddots & \vdots \\ N_{b\_t,p,1} & N_{b\_t,p,2} & \cdots & N_{b\_t,p,n} \end{bmatrix}$ denotes the design matrix.
And we have

$$
\begin{aligned}
& tr\Big\{\tilde{\mathbf{W}}_t^{\mathrm{T}}\cdot\Big[4\hat{\mathbf{V}}_t\mathbf{P}\mathbf{e}(t)\mathbf{x}_t^{\mathrm{T}}+\dot{\hat{\mathbf{W}}}_t\Big]\Big\}=tr\Big\{\tilde{\mathbf{W}}_t^{\mathrm{T}}\mathbf{N}_{w\_t}\mathbf{e}^{\mathrm{T}}(t)\Big\} \\
& = tr\left\{\begin{bmatrix} \tilde{W}_{t,1,1} & \tilde{W}_{t,1,2} & \cdots & \tilde{W}_{t,1,n} \\ \tilde{W}_{t,2,1} & \tilde{W}_{t,2,2} & \cdots & \tilde{W}_{t,2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{W}_{t,p,1} & \tilde{W}_{t,p,2} & \cdots & \tilde{W}_{t,p,n} \end{bmatrix}^{\mathrm{T}}\cdot\begin{bmatrix} N_{w\_t,1} \\ N_{w\_t,2} \\ \vdots \\ N_{w\_t,p} \end{bmatrix}\cdot\begin{bmatrix} e_1(t) \\ e_2(t) \\ \vdots \\ e_n(t) \end{bmatrix}^{\mathrm{T}}\right\} \\
& = \sum_{i=1}^{n}\Big[\tilde{\mathbf{W}}_{t,\bullet,i}^{\mathrm{T}}\cdot\mathbf{N}_{w\_t}\cdot e_i(t)\Big]
\end{aligned}
$$

$$
\begin{aligned}
&= \begin{bmatrix} \tilde{\mathbf{W}}_{t,\bullet,1} \\ \tilde{\mathbf{W}}_{t,\bullet,2} \\ \vdots \\ \tilde{\mathbf{W}}_{t,\bullet,n} \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} \mathbf{N}_{w\_t} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_{w\_t} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{N}_{w\_t} \end{bmatrix} \cdot \begin{bmatrix} e_1(t) \\ e_2(t) \\ \vdots \\ e_n(t) \end{bmatrix} \\
&= \mathbf{J}_{w\_t}^{\mathrm{T}} \cdot \mathbf{\Omega}_{w\_t} \cdot \mathbf{e}(t), \\
&\sum_{i=1}^{t-1} \left[tr\left\{ \tilde{\mathbf{W}}_i^{\mathrm{T}} \cdot \left[4\hat{\mathbf{V}}_t \mathbf{P}\mathbf{e}(t)\mathbf{x}_i^{\mathrm{T}} + \dot{\hat{\mathbf{W}}}_i \right] \right\} \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j) \right] \\
&= \sum_{i=1}^{t-1} \left[\mathbf{J}_{w\_i}^{\mathrm{T}} \cdot \mathbf{\Omega}_{w\_i} \cdot \mathbf{e}(t) \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j) \right], \\
&tr\left\{ \tilde{\mathbf{V}}_t^{\mathrm{T}} \cdot \left[2\hat{\mathbf{S}}_t \mathbf{e}^{\mathrm{T}}(t)\mathbf{P} + \dot{\hat{\mathbf{V}}}_t \right] \right\} - 2\epsilon_2^{\mathrm{T}} \tilde{\mathbf{V}}_t \mathbf{P}\mathbf{e}(t) = tr\left\{ \tilde{\mathbf{V}}_t^{\mathrm{T}} \mathbf{N}_{v\_t} \mathbf{e}^{\mathrm{T}}(t) \right\} - 2\epsilon_2^{\mathrm{T}} \tilde{\mathbf{V}}_t \mathbf{P}\mathbf{e}(t) \\
&= \sum_{i=1}^{n} \left[\tilde{\mathbf{V}}_{t,\bullet,i}^{\mathrm{T}} \cdot \mathbf{N}_{v\_t} \cdot e_i(t) \right] - 2\sum_{i=1}^{n} \left[\tilde{\mathbf{V}}_{t,\bullet,i}^{\mathrm{T}} \cdot \epsilon_2 \cdot \mathbf{P}_{i\bullet} \cdot \mathbf{e}_(t) \right] \\
&= \begin{bmatrix} \tilde{\mathbf{V}}_{t,\bullet,1} \\ \tilde{\mathbf{V}}_{t,\bullet,2} \\ \vdots \\ \tilde{\mathbf{V}}_{t,\bullet,n} \end{bmatrix}^{\mathrm{T}} \cdot \left(\begin{bmatrix} \mathbf{N}_{v\_t} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_{v\_t} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{N}_{v\_t} \end{bmatrix} - 2 \begin{bmatrix} \epsilon_2 \cdot \mathbf{P}_{1\bullet} \\ \epsilon_2 \cdot \mathbf{P}_{2\bullet} \\ \vdots \\ \epsilon_2 \cdot \mathbf{P}_{n\bullet} \end{bmatrix} \right) \cdot \begin{bmatrix} e_1(t) \\ e_2(t) \\ \vdots \\ e_n(t) \end{bmatrix} \\
&= \mathbf{J}_{v\_t}^{\mathrm{T}} \cdot \mathbf{\Omega}_{v\_t} \cdot \mathbf{e}(t), \\
&\tilde{\mathbf{b}}_t^{\mathrm{T}} \cdot \left[4\hat{\mathbf{V}}_t \mathbf{P}\mathbf{e}(t) + \dot{\hat{\mathbf{b}}}_t \right] = \tilde{\mathbf{b}}_t^{\mathrm{T}} \cdot \mathbf{N}_{b\_t} \cdot \mathbf{e}(t).
\end{aligned}
$$

The derivative of V (t; e(t)) can be obtained as follows

$$
\begin{aligned}
\dot{V}(t, \mathbf{e}(t)) \leqslant{}& \mathbf{e}^{\mathrm{T}}(t)\left[-(\mathbf{A} + \mathbf{L}\mathbf{H})^{\mathrm{T}}\mathbf{P} - \mathbf{P}(\mathbf{A} + \mathbf{L}\mathbf{H}) \right]\mathbf{e}(t) - \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\epsilon_1 - \epsilon_1^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
&+ \mathbf{J}_{w\_t}^{\mathrm{T}} \cdot \mathbf{\Omega}_{w\_t} \cdot \mathbf{e}(t) + \sum_{i=1}^{t-1} \left[\mathbf{J}_{w\_i}^{\mathrm{T}} \cdot \mathbf{\Omega}_{w\_i} \cdot \mathbf{e}(t) \cdot \prod_{j=i+1}^{t} \sigma(\mathbf{x}_j) \right] + \mathbf{J}_{v\_t}^{\mathrm{T}} \cdot \mathbf{\Omega}_{v\_t} \cdot \mathbf{e}(t) \\
&+ \tilde{\mathbf{b}}_t^{\mathrm{T}} \cdot \mathbf{N}_{b\_t} \cdot \mathbf{e}(t) \\
\leqslant{}& \mathbf{e}^{\mathrm{T}}(t)\left[-(\mathbf{A} + \mathbf{L}\mathbf{H})^{\mathrm{T}}\mathbf{P} - \mathbf{P}(\mathbf{A} + \mathbf{L}\mathbf{H}) \right]\mathbf{e}(t) - \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\epsilon_1 - \epsilon_1^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
&+ \begin{bmatrix} \mathbf{J}_{w\_t} \\ \mathbf{J}_{w\_t-1} \\ \vdots \\ \mathbf{J}_{w\_1} \\ \mathbf{J}_{v\_t} \\ \tilde{\mathbf{b}}_t \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} \mathbf{\Omega}_{w\_t} \\ \mathbf{\Omega}_{w\_t-1} \cdot \sigma(\mathbf{x}_t) \\ \vdots \\ \mathbf{\Omega}_{w\_1} \cdot \prod_{j=2}^{t} \sigma(\mathbf{x}_j) \\ \mathbf{\Omega}_{v\_t} \\ \mathbf{N}_{b\_t} \end{bmatrix} \cdot \mathbf{e}(t) \\
\leqslant{}& \mathbf{e}^{\mathrm{T}}(t)\left[-(\mathbf{A} + \mathbf{L}\mathbf{H})^{\mathrm{T}}\mathbf{P} - \mathbf{P}(\mathbf{A} + \mathbf{L}\mathbf{H}) \right]\mathbf{e}(t) - \mathbf{e}^{\mathrm{T}}(t)\mathbf{P}\epsilon_1 - \epsilon_1^{\mathrm{T}}\mathbf{P}\mathbf{e}(t) \\
&+ \mathbf{J}_{wvb}^{\mathrm{T}} \cdot \mathbf{\Omega}_{wvb} \cdot \mathbf{e}(t) \\
\leqslant{}& \begin{bmatrix} \mathbf{e}(t) \\ \epsilon_1 \\ \mathbf{J}_{wvb} \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} \mathbf{\Omega}_3 & -\mathbf{P} & \frac{1}{2}\mathbf{\Omega}_{wvb}^{\mathrm{T}} \\ -\mathbf{P} & \mathbf{0} & \mathbf{0} \\ \frac{1}{2}\mathbf{\Omega}_{wvb} & \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{e}(t) \\ \epsilon_1 \\ \mathbf{J}_{wvb} \end{bmatrix} \\
\leqslant{}& \mathbf{J}_1^{\mathrm{T}} \cdot \begin{bmatrix} \mathbf{\Omega}_3 & -\mathbf{P} & \frac{1}{2}\mathbf{\Omega}_{wvb}^{\mathrm{T}} \\ -\mathbf{P} & \mathbf{0} & \mathbf{0} \\ \frac{1}{2}\mathbf{\Omega}_{wvb} & \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \mathbf{J}_1.
\end{aligned}
$$

Select appropriate $\mathbf{N}_{w\_i}$, $\mathbf{N}_{v\_t}$ and $\mathbf{N}_{b\_t}$ to satisfy

$$\begin{bmatrix} \mathbf{\Omega}_3 & -\mathbf{P} & \frac{1}{2}\mathbf{\Omega}_{wvb}^{\mathrm{T}} \\ -\mathbf{P} & \mathbf{0} & \mathbf{0} \\ \frac{1}{2}\mathbf{\Omega}_{wvb} & \mathbf{0} & \mathbf{0} \end{bmatrix} \leqslant 0,$$

then the error system (14) is asymptotically stable, and the proof of Theorem 3 is completed. □

Appendix D. The Proof of Theorem 4

Proof. By using Assumption 3 and (14), we have

$$\dot{\hat{\mathbf{y}}}(t) - \dot{\mathbf{y}}(t) = \mathbf{H}\big(\dot{\hat{\mathbf{x}}}(t) - \dot{\mathbf{x}}(t)\big) = \mathbf{H}\big[-(\mathbf{A}+\mathbf{LH})\mathbf{e}(t) + \big(\hat{\mathbf{K}} - \bar{\mathbf{K}} - \epsilon_1\big)\big].$$

By using (17), Theorems 2 and 3, we have

$$\begin{bmatrix} \hat{\mathbf{y}}(t) - \mathbf{y}(t) \\ \dot{\hat{\mathbf{y}}}(t) - \dot{\mathbf{y}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{H} \\ -\mathbf{H}(\mathbf{A}+\mathbf{LH}) \end{bmatrix} \mathbf{e}(t).$$

And we have

$$\mathbf{Y} = \mathbf{G} \cdot \mathbf{e}(t),$$

where $\mathbf{G} = \begin{bmatrix} \mathbf{H} \\ -\mathbf{H}(\mathbf{A}+\mathbf{LH}) \end{bmatrix}$ and $\mathbf{Y} = \begin{bmatrix} \hat{\mathbf{y}}(t) - \mathbf{y}(t) \\ \dot{\hat{\mathbf{y}}}(t) - \dot{\mathbf{y}}(t) \end{bmatrix}$.

If $\mathbf{G}$ is left invertible, there exists $\mathbf{G}^{-1}$ such that $\mathbf{e}(t) = \mathbf{G}^{-1} \cdot \mathbf{Y}$, and the proof of Theorem 4 is completed. □

References

1. Chua, L. Memristor-the missing circuit element. *IEEE Trans. Circuit Theory* **1971**, *18*, 507–519. [CrossRef]
2. Wu, A.; Wen, S.; Zeng, Z. Synchronization control of a class of memristor-based recurrent neural networks. *Inf. Sci.* **2012**, *183*, 106–116. [CrossRef]
3. Wen, S.; Zeng, Z.; Huang, T. Exponential stability analysis of memristor-based recurrent neural networks with time-varying delays. *Neurocomputing* **2012**, *97*, 233–240. [CrossRef]
4. Chen, J.; Zeng, Z.; Jiang, P. Global mittag-leffler stability and synchronization of memristor-based fractional-order neural networks. *Neural Netw.* **2014**, *51*, 1–8. [CrossRef] [PubMed]
5. Wang, F.; Na, H.; Wu, S.; Yang, X.; Guo, Y.; Lim, G.; Rashid, M.M. Delayed switching applied to memristor neural networks. *J. Appl. Phys.* **2012**, *111*, 507–511. [CrossRef]
6. Jiang, M.; Mei, J.; Hu, J. New results on exponential synchronization of memristor-based chaotic neural networks. *Neurocomputing* **2015**, *156*, 60–67. [CrossRef]
7. Wu, H.; Li, R.; Ding, S.; Zhang, X.; Yao, R. Complete periodic adaptive antisynchronization of memristor-based neural networks with mixed time-varying delays. *Can. J. Phys.* **2014**, *92*, 1337–1349. [CrossRef]
8. Hu, M.; Chen, Y.; Yang, J.; Wang, Y.; Li, H.H. A compact memristor-based dynamic synapse for spiking neural networks. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2016**, *36*, 1353–1366. [CrossRef]
9. Zheng, M.; Li, L.; Xiao, J.; Yang, Y.; Zhao, H. Finite-time projective synchronization of memristor-based delay fractional-order neural networks. *Nonlinear Dyn.* **2017**, *89*, 2641–2655. [CrossRef]
10. Negrov, D.; Karandashev, I.; Shakirov, V.; Matveyev, Y.A. A plausible memristor implementation of deep learning neural networks. *Neurocomputing* **2015**, *237*, 193–199. [CrossRef]
11. Wang, X.; Ju, H.; Yang, H.; Zhong, S. A new settling-time estimation protocol to finite-time synchronization of impulsive memristor-based neural networks. *IEEE Trans. Cybern.* **2020**, *52*, 4312–4322. [CrossRef] [PubMed]
12. Chen, B.; Yang, H.; Zhuge, F.; Li, Y.; Chang, T.-C.; He, Y.-H.; Yang, W.; Xu, N.; Miao, X.-S. Optimal tuning of memristor conductance variation in spiking neural networks for online unsupervised learning. *IEEE Trans. Electron Devices* **2019**, *66*, 2844–2849. [CrossRef]
13. Liu, X.Y.; Zeng, Z.G.; Wunsch, D.C. Memristor-based LSTM network with in situ training and its applications. *Neural Netw.* **2020**, *131*, 300–311. [CrossRef] [PubMed]
14. Ning, L.; Wei, X. Bipartite synchronization for inertia memristor-based neural networks on coopetition networks. *Neural Netw.* **2020**, *124*, 39–49.
15. Xu, C.; Wang, C.; Sun, Y.; Hong, Q.; Deng, Q.; Chen, H. Memristor-based neural network circuit with weighted sum simultaneous perturbation training and its applications. *Neurocomputing* **2021**, *462*, 581–590. [CrossRef]

16. Prezioso, M.; Bayat, F.M.; Hoskins, B.D.; Likharev, K.K.; Strukov, D.B. Training andoperation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **2015**, *521*, 61–64. [CrossRef] [PubMed]
17. Kim, S.; Park, J.; Kim, T.H.; Hong, K.; Hwang, Y.; Park, B.g.; Kim, H. 4-bit multilevel operation in overshoot suppressed Al2O3/TiOx resistive random-access memory crossbar array. *Adv. Intell. Syst.* **2022**, *4*, 2100273. [CrossRef]
18. Wang, Z.R.; Joshi, S.; Savel'ev, S.; Song, W.; Midya, R.; Li, Y.; Rao, M.; Yan, P.; Asapu, S.; Zhuo, Y.; et al. Fully memristive neural networks for pattern classification with unsupervised learning. *Nat. Electron.* **2018**, *1*, 137–145. [CrossRef]
19. Choi, W.S.; Jang, J.T.; Kim, D.; Yang, T.J.; Kim, C.; Kim, H.; Kim, D.H. Influence of Al2O3 layer on InGaZnO memristor crossbar array for neuromorphic applications. *Chaos Solitons Fractals* **2022**, *156*, 111813. [CrossRef]
20. Bayat, F.M.; Prezioso, M.; Chakrabarti, B.; Nili, H.; Kataeva, I.; Strukov, D.B. Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits. *Nat. Commun.* **2018**, *9*, 2331. [CrossRef]
21. Chen, Y.; Chen, G. Stability analysis of delayed neural networks based on a relaxed delay-product-type lyapunov functional. *Neurocomputing* **2021**, *439*, 340–347. [CrossRef]
22. Wang, L.; Song, Q.; Liu, Y.; Zhao, Z.; Alsaadi, F.E. Finite-time stability analysis of fractional-order complex-valued memristor-based neural networks with both leakage and time-varying delays. *Neurocomputing* **2017**, *245*, 86–101. [CrossRef]
23. Meng, Z.; Xiang, Z. Stability analysis of stochastic memristor-based recurrent neural networks with mixed time-varying delays. *Neural Comput. Appl.* **2017**, *28*, 1787–1799. [CrossRef]
24. Chen, C.; Zhu, S.; Wei, Y. Finite-time stability of delayed memristor-based fractional-order neural networks. *IEEE Trans. Cybern.* **2020**, *50*, 1607–1616. [CrossRef]
25. Du, F.; Lu, J. New criteria for finite-time stability of fractional order memristor-based neural networks with time delays. *Neurocomputing* **2021**, *421*, 349–359. [CrossRef]
26. Rakkiyappan, R.; Chandrasekar, A.; Laksmanan, S.; Park, J.H. State estimation of memristor-based recurrent neural networks with time-varying delays based on passivity theory. *Complexity* **2014**, *19*, 32–43. [CrossRef]
27. Bao, H.; Cao, J.; Kruths, J.; Alsaedi, A.; Ahmad, B. H∞ state estimation of stochastic memristor-based neural networks with time-varying delays. *Neural Netw.* **2018**, *99*, 79–91. [CrossRef]
28. Nagamani, G.; Rajan, G.S.; Zhu, Q. Exponential state estimation for memristor-based discrete-time bam neural networks with additive delay components. *IEEE Trans. Cybern.* **2020**, *5*, 4281–4292. [CrossRef]
29. Sakthivel, R.; Anbuvithya, R.; Mathiyalagan, K.; Prakash, P. Combined h∞ and passivity state estimation of memristive neural networks with random gain fluctuations. *Neurocomputing* **2015**, *168*, 1111–1120. [CrossRef]
30. Wei, F.; Chen, G.; Wang, W. Finite-time synchronization of memristor neural networks via interval matrix method. *Neural Netw.* **2020**, *127*, 7–18. [CrossRef]
31. Wang, J.; Wang, Z.; Chen, X.; Qiu, J. Synchronization criteria of delayed inertial neural networks with generally markovian jumping. *Neural Netw.* **2021**, *139*, 64–76. [CrossRef] [PubMed]
32. Li, L.; Xu, R.; Gan, Q.; Lin, J. Synchronization of neural networks with memristor-resistor bridge synapses and lévy noise. *Neurocomputing* **2021**, *432*, 262–274. [CrossRef]
33. Ren, H.; Peng, Z.; Gu, Y. Fixed-time synchronization of stochastic memristor-based neural networks with adaptive control. *Neural Netw.* **2020**, *130*, 165–175. [CrossRef] [PubMed]
34. Zheng, C.D.; Zhang, L. On synchronization of competitive memristor-based neural networks by nonlinear control. *Neurocomputing* **2020**, *410*, 151–160. [CrossRef]
35. Pan, C.; Bao, H. Exponential synchronization of complex-valued memristor-based delayed neural networks via quantized intermittent control. *Neurocomputing* **2020**, *404*, 317–328. [CrossRef]
36. Xiao, J.; Li, Y.; Zhong, S.; Xu, F. Extended dissipative state estimation for memristive neural networks with time-varying delay. *ISA Trans.* **2016**, *64*, 113–128. [CrossRef] [PubMed]
37. Li, R.; Gao, X.; Cao, J.; Zhang, K. Dissipativity and exponential state estimation for quaternion-valued memristive neural networks. *Neurocomputing* **2019**, *363*, 236–245. [CrossRef]
38. Li, H. Sampled-data state estimation for complex dynamical networks with time-varying delay and stochastic sampling. *Neurocomputing* **2014**, *138*, 78–85. [CrossRef]
39. Dai, X.; Yin, H.; Jha, N. Grow and prune compact, fast, and accurate lstms. *IEEE Trans. Comput.* **2020**, *69*, 441–452. [CrossRef]
40. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366. [CrossRef]