A New Supervised Learning Algorithm of Recurrent Neural Networks and L_2 Stability Analysis in Discrete-Time Domain

Wu Yilei, Yang Xulei and Song Qing
School of Electrical and Electronic Engineering
Nanyang Technological University,
Singapore

1. Introduction

In the past decades, Recurrent Neural Network (RNN) has attracted extensive research interests in various disciplines. One important motivation of these investigations is the RNN's promising ability of modeling time-behavior of nonlinear dynamic systems. It has been theoretically proved that RNN is able to map arbitrary input sequences to output sequences with infinite accuracy regardless underline dynamics with sufficient training samples [1]. Moreover, from biological point of view, RNN is more plausible to the real neural models as compared to other adaptive methods such as Hidden Markov Models (HMM), feed-forward networks and Support Vector Machines (SVM). From the practical point of view, the dynamics approximation and adaptive learning capability make RNN a highly competitive candidate for a wide range of applications. See [2] [3] [4] for examples.

Among the various applications, the realtime signal processing has constantly been one of the active topics of RNN. In such kind of applications, the convergence speed is always an important concern because of the tight timing requirement. For example, the conventional training algorithms of RNN, such as the Backpropagation Through Time (BPTT) and the Real Time Recurrent Learning (RTRL) always suffer from slow convergence speed. If a large learning rate is selected to speed up the weight updating, the training process may become unstable. Thus it is desirable to develop robust learning algorithms with variable or adaptive learning coe±cients to obtain a tradeoff between the stability and fast convergence speed.

The issue has already been extensively studied for linear adaptive filters, e.g., the famous Normalized Least Mean Square (N-LMS) algorithm. However, for online training algorithms of RNN this is still an open topic. Due to the inherent feedback and distributive parallel structure, the adjustments of RNN weights can affect the entire neural network state variables during network training. Hence it is difficult to obtain the error derivative for gradient type updating rules, and in turn difficulty in the analysis of the underlying dynamics of the training. So far, a great number of works have been carried out to solve the problem. To name a few, in [5], B. Pearlmutter presented a detail survey on gradient calculation for RNN training algorithms. In [6] [7] , M. Rupp et al introduced a robustness

Source: Recurrent Neural Networks, Book edited by: Xiaolin Hu and P. Balasubramaniam, ISBN 978-953-7619-08-4, pp. 400, September 2008, I-Tech, Vienna, Austria

analysis of RNN by the small gain theorem. The stability was explained from the energy point of view that the ratio of output noise against input noise was guaranteed to be smaller than unity. In [8], J. Liang and M. Gupta studied the stability of dynamic back-propagation training algorithm by the Lyapunov method. An auxiliary term was appended to augment the learning error. The convergence speed was improved by introducing an extra increment in the updating rule. Later, A. Atiya and A. Parlos used a generalized steepest descent method to obtain a unified error gradient algorithm [9]. Recently, Q. Song et al proposed a simultaneous perturbation stochastic approximation training method for neural networks and robust stability is established by the conic sector theorem [10] [11].

The work presented in this chapter investigate the stability and robustness of the gradient-type training algorithms of RNN in the discrete-time domain. A Robust Adaptive Gradient Descent (RAGD) training algorithm is introduced to improve the RNN training speed as compared to those conventional algorithms, such as the BPTT, the RTRL and the Normalized RTRL (N-RTRL). The main feature of the RAGD is the novel hybrid training concept, which switches the training patterns between the standard online Back Propagation (BP) and the N-RTRL algorithm via three adaptive parameters, the hybrid adaptive learning rates, the adaptive dead zone learning rates, and the normalization factors. These parameters allow RAGD to locate relatively deeper local attractors of the training and hence obtain a faster transient response. Different from the N-RTRL, the RAGD uses a specifically designed error derivatives based on the extended recurrent gradient to approximate the true gradient for realtime learning. Also the RAGD is different from the static BP in terms that the former uses the extended recurrent gradient to extend the instantaneous squared estimation error minimization into recurrent mode, while the latter is strictly based on the instantaneous squared estimation error minimization without specifically considering the recurrent signal.

Weight convergence and robust stability of the RAGD are proved respectively based on the Lyapunov function and the Cluett's law, which is developed from the conic sector theorem of input- output system theory. Sufficient boundary conditions of the three adaptive parameters are derived to guarantee the L_2 stability of the training. Different from precedent results [12], the present work employs the input-output systematic approach in analysis. This is because the input-output theory on basis of functional analysis requires minimal assumptions about the training statistics. Although the results are also derivable from conventional analysis method, we emphasize that input-output systematic scheme can provide an in-depth understanding of RNN training dynamics from different aspect.

In addition to the theoretical analysis, we carried out three case studies of the applications in realtime signal processing via computer simulations, including time series prediction, system identification, and attractor learning for pattern association. With these case studies, we are able to qualify the effectiveness of the RAGD and hence justify that the algorithm outperforms other counterparts.

The overall chapter is organized as follows: In Sections 2, we briefly introduce the structure of the RNN and the RAGD training algorithm. In Section 3, the robustness analysis of the RAGD is carried out for the Single-input Single-Output and Multi-input Multi-output RNN respectively. In addition, the conic sector theorem is introduced as the theoretical foundation of the analysis. Computer simulations are presented in Section 4 to show the efficiency of our proposed RAGD. Section 5 draws the final conclusions.

2. RAGD learning algorithm

Consider a RNN with *l* output nodes and *m* hidden neurons. In discrete-time domain, the network output $\hat{y}$ at time instant *k* can be written as

$$\hat{y}(k) = \hat{V}(k)\Phi(\hat{W}(k)\hat{x}(k)) \tag{1}$$

where $\hat{V}$ $(k) \in R^{l\times m}$ and $\hat{W}$ $(k) \in R^{m\times n}$ are output and hidden layer weights respectively (in matrix form), $\Phi(\cdot) \in R^{m\times 1}$ is a vector of nonlinear activation functions, and $\hat{x}$ $(k) \in R^{n\times 1}$ is the state vector that consists of external input $u(k)$ and n - 1 delayed output feedback entries

$$\hat{x}(k) = [u(k), \hat{y}(k-1), \cdots, \hat{y}(k-n+1)]^T \tag{2}$$

in which *T* denotes transpose operation. To simplify the expression, we use notation Φ (k) instead of Φ $(\hat{W}$ (k) $\hat{x}$ $(k))$ hereafter. When estimating a command signal $d(k)$, the instantaneous modeling error of RNN can be defined by

$$e(k) = d(k) - \hat{y}(k) + \varepsilon(k) \tag{3}$$

Note a disturbance term ε $(k) \in R^{l\times 1}$ is taken into account in (3). Without loss of generality, there is no assumption on the prior knowledge of ε (k) and its statistics. The training objective of RNN is to update the weight parameters step by step to minimize certain cost function $f(e(k))$, with the most convenient form being the squared instantaneous error $e^2(k)/2$. Specifically, in an environment of time-varying signal statistics, a gradient based sequential training algorithm can be used to recursively reduce the $f(e(k))$ by estimating the weights at each time instant

$$\begin{cases} \hat{V}(k+1) = \hat{V}(k) - \alpha \frac{\partial f(e(k))}{\partial \hat{V}(k)} \\ \hat{W}(k+1) = \hat{W}_i(k) - \alpha \frac{\partial f(e(k))}{\partial \hat{W}_i(k)} \end{cases} \tag{4}$$

where α is the learning rate of RNN, and $\hat{W}$ $_i(k)$ is the *i*th row of hidden layer weight matrix, with i = 1, 2, …, m. Note subscript *i* denotes *i*th row for matrices or *i*th entry for vectors. As for the above algorithm, a widely recognized problem is the slow convergence speed because of small learning rates for purpose of preserving weight convergence. So far the commonly accepted solution of this problem is to employ normalization, e.g., the N-RTRL algorithm [13] [1]. Indeed, the solution can be further improved if we can find effective boundary conditions of learning rates and normalization factors as will be shown in later sections. Moreover, hybrid learning rates can be employed to obtain the tradeoff between the transient and steady state response. Now based on the RNN model (1) and the gradient-based training equation (4), we propose the RAGD learning algorithm as follows

$$\begin{cases} \hat{V}(k+1) = \hat{V}(k) + \frac{\alpha^v(k)}{\rho^v(k)} e(k)(\Phi(k)^T + \beta^v(k)\hat{A}(k)) \\ \hat{W}(k+1) = \hat{W}(k) + \frac{\alpha^w(k)}{\rho^w(k)} diag\{\Phi'(k)\}\hat{V}(k)^T e(k)(\hat{x}(k)^T + \beta^w(k)\hat{B}(k)) \end{cases} \tag{5}$$

where $\Phi'(k)$ is the vector of activation function derivatives, $\alpha^v(k)$, $\alpha^w(k)$ are adaptive dead zone learning rates, $\beta^v(k)$, $\beta^w(k)$ are hybrid learning rates, $\rho^v(k)$, $\rho^w(k)$ are normalization factors, and $\hat{A}(k)$, $\hat{B}(k)$ are residual error gradients. These variables are defined in the following.

(a) $\Phi'(k) \in R^{m\times 1}$

$$\Phi'(k) = [\ \phi'(\hat{W}_1(k)\hat{x}(k)) \quad \phi'(\hat{W}_2(k)\hat{x}(k)) \quad \cdots \quad \phi'(\hat{W}_m(k)\hat{x}(k))\ ]^T \tag{6}$$

(b) $\hat{A}(k) \in R^{1\times m}$ and $\hat{B}(k) \in R^{1\times n}$

$$\hat{A}(k) \quad = \quad \underline{\hat{V}}(k)\cdot[diag\{\Phi'(k)\}]_l \cdot \left[\hat{W}(k)\right]_l \cdot \hat{D}^v(k) \tag{7}$$

$$\hat{B}(k) \quad = \quad \underline{\hat{W}}(k)\hat{D}^w(k) \tag{8}$$

where $[diag\{\Phi'(k)\}]_l \in R^{(l\times m)\times(l\times m)}$ and $\left[\hat{W}(k)\right]_l \in R^{(l\times m)\times(l\times n)}$ are block diagonal matrices with sub-matrix $diag\{\Phi'(k)\}$ and $\hat{W}(k)$ on the diagonal respectively

$$[diag\{\Phi'(k)\}]_l = \begin{bmatrix} diag\{\Phi'(k)\} & & & 0 \\ & diag\{\Phi'(k)\} & & \\ & & \ddots & \\ 0 & & & diag\{\Phi'(k)\} \end{bmatrix}$$

$$\left[\hat{W}(k)\right]_l = \begin{bmatrix} \hat{W}(k) & & & 0 \\ & \hat{W}(k) & & \\ & & \ddots & \\ 0 & & & \hat{W}(k) \end{bmatrix}$$

$\underline{\hat{V}}(k) \in R^{1\times(l\times m)}$ and $\underline{\hat{W}}(k) \in R^{1\times(m\times n)}$are long vector versions of the weight matrices $\hat{V}(k)$ and $\hat{W}(k)$ respectively

$$\begin{cases} \underline{\hat{V}}(k) = \left[\hat{V}_1(k) \quad \hat{V}_2(k) \quad \cdots \quad \hat{V}_l(k)\right] \\ \underline{\hat{W}}(k) = \left[\hat{W}_1(k) \quad \hat{W}_2(k) \quad \cdots \quad \hat{W}_m(k)\right] \end{cases}$$

and the Jacobian $\hat{D}^v(k) \in R^{(l\times n)\times m}$ and $\hat{D}^w(k) \in R^{(m\times n)\times n}$

$$\begin{cases} \hat{D}^v(k) = [\hat{D}_1^v(k)^T \quad \hat{D}_2^v(k)^T \quad \cdots \quad \hat{D}_l^v(k)^T]^T \\ \hat{D}^w(k) = [\hat{D}_1^w(k)^T \quad \hat{D}_2^w(k)^T \quad \cdots \quad \hat{D}_m^w(k)^T]^T \end{cases}$$

in which $\hat{D}_i^v(k) = \frac{\partial \hat{x}(k)}{\partial \hat{V}_i(k)} \in R^{n\times m}$, $\hat{D}_i^w(k) = \frac{\partial \hat{x}(k)}{\partial \hat{W}_i(k)} \in R^{n\times n}$ are sub-matrices.

(c) $\beta^v(k)$ and $\beta^w(k)$

$$\beta^v(k) = sgn\{\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T\} \tag{9}$$

$$\beta^w(k) = sgn\{\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T\} \tag{10}$$

where δ is a small positive constant, *I* is the identity matrix, and δ*I* is employed to ensure the matrix $\delta I + \Phi(k)\Phi(k)^T$ and $\delta I + \hat{x}(k)\hat{x}(k)^T$ positive definite.

(d) $\rho^v(k)$ and $\rho^w(k)$

$$\rho^v(k) = \nu\rho^v(k-1) + \max\{\bar{\rho}^v, \|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2\} \tag{11}$$

$$\rho^w(k) = \nu\rho^w(k-1) + \max\{\bar{\rho}^w, \frac{\mu_{max}\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2}{\phi'_{min}(k)}\} \tag{12}$$

where $\nu < 1$, $\bar{\rho}^v$ and $\bar{\rho}^w < 1$ are positive constants, μ_{max} is the maximu value of the activation function, and $\phi'_{min}(k) = \min\{\Phi'_1(k), \cdots, \Phi'_m(k)\}$. Note we are using an inner product induced norm, the Frobenius norm, as the norm of weight matrices in this work.

(e) $\alpha^v(k)$ and $\alpha^w(k)$

$$\alpha^v(k) = sgn\{\|e(k)\| - \varepsilon^v_{max}/\sqrt{1 - \frac{\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)}}\} \tag{13}$$

$$\alpha^w(k) = sgn\{\|e(k)\| - \varepsilon^w_{max}/\sqrt{1 - \frac{\mu_{max}\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2}{\phi'_{min}(k) \cdot \rho^w(k)}}\} \tag{14}$$

where $\varepsilon^v_{max} = \max\{\|\tilde{\varepsilon}^v(k)\|\}$, $\varepsilon^w_{max} = \max\{\|\tilde{\varepsilon}^w(k)\|\}$, and *sgn*(•) function is defined by

$$sgn(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{15}$$

Remark 1 The RAGD algorithm uses the specific designed derivative as shown in (5). The state estimators are taken into account in the second terms of the partial derivatives on the right side of the equation. Further, to make the proposed algorithm realtime adaptive and recurrent, the $\hat{D}^v(k)$ *and the* $\hat{D}^w(k)$ *in the partial derivatives are calculated on basis of the data from previous training steps, which is similar to that of the N-RTRL algorithm [14]. It is noteworthy only when the convergence and stability requirements (details will be given in Section 3) are met, they hybrid learning rate* β *will be turned on. In this case, since we have estimated the best available gradient at each step k, the combination of weights and state estimates in (5) should provide a relatively deeper local attractor of the nonlinear iteration, and hence to speed up the training.*

3. Robust stability analysis

In this section, we present detail analysis of robust stability of the RAGD algorithm. Proofs of weight convergence and L_2 stability are derived on basis of Lyapunov function and input-output systematic approach respectively. The boundary conditions on the three adaptive parameters, the hybrid learning rate, the adaptive dead zone learning rates, and the normalization factors, are obtained for the optimized transient response of the training. For better understanding of the algorithm, a simple case of Single-input Single-output (SISO) RNN is firstly given as an example. Then the results are extended to the more complicated case of Multi-input Multi-output (MIMO) RNN. Before proceeding, we introduce the Cluett's law and mathematical preliminaries.

3.1 Cluett's laws

The main concern of this work is discrete signals which are infinite sequences of real numbers. Each signal may be considered an element of a set known as a linear vector space. To provide a clear explanation, an immediate review is given on several mathematical notations. Let the $x(k) \in R^{n\times 1}$ denotes the series $\{x(1), x(2), \ldots\}$, then

i) The L_2 norm of $x(k)$ is defined as $\|x(k)\|_2 = \sqrt{\sum_{k=1}^{\infty} \|x(k)\|^2}$

ii) If the L_2 norm of $x(k)$ exists, the corresponding normed vector spaces are called L_2 spaces;

iii) The truncation of $x(k)$ is defined as $\|x(k)\|_{2,N} = \sqrt{\sum_{k=1}^{N} \|x(k)\|^2}$

iv) The extension of a space L_2, denoted by L_{2e} is the space consisting of those elements $x(k)$ whose truncations are all lie in L_2, i.e., $\|x(k)\|_{2,N} < \infty$, for all $N \in Z_+$ (the set of positive integers).

Note $\|\bullet\|$ denotes the Euclidean norm of a vector, and $\|\bullet\|_2$ for the L_2 norm of a signal (could be either a vector or a scalar). Let's consider the closed loop system shown in Figure 1

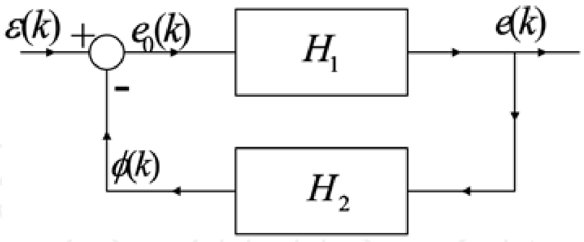


Figure 1. A general closed loop feedback system

$$\begin{cases} e_0(k) = \varepsilon(k) - \phi(k) \\ e(k) = H_1 e_0(k) \\ \phi(k) = H_2 e(k) \end{cases} \tag{16}$$

where operators $H_1;H_2 : L_{2e} \to L_{2e}$, discrete time signals $e_0(k); e(k); \phi(k) \in L_{2e}$ and $\varepsilon(k) \in L_2$.

Theorem 1 *(Cluett's Law-1) If the following two conditions hold* *i)* $H_1 : e_0(k) \rightarrow e(k)$ *satisfies* $\sum_{k=0}^{N} [e^2(k) + \alpha e_0(k)e(k) + \beta e_0^2(k)] \geq -\gamma,\ \forall N \in Z^+$

ii) $H_2 : e(k) \rightarrow \phi(k)$ *satisfies* $\sum_{k=0}^{N} [\beta\phi^2(k) - \alpha\phi(k)e(k) + e^2(k)] \leq -\eta\{\|(\phi(k), e(k))\|_2^2\}_N,\ \forall N \in Z^+$

for some α, $\beta \in R$, which are independent of k and N, and $\gamma \geq 0$, $\eta > 0$, which are independent of N, then the closed loop feedback system of (16) is stable in the sense of $e(k)$, $\phi(k) \in L_2$.

Proof: By the inequality i) and using $e_0(k) = \varepsilon(k) - \phi(k)$

$$\sum_{k=0}^{N}[\beta\phi^2(k) - \alpha\phi(k)e(k) + e^2(k)] + \sum_{k=0}^{N}[\alpha\varepsilon(k)e(k) - 2\beta\varepsilon(k)\phi(k) + \beta\varepsilon^2(k)] \geq -\gamma \quad (17)$$

Combining inequality ii) and equation (17)

$$-\eta\{\|(\phi(k), e(k))\|_2^2\}_N + \sum_{k=0}^{N}[\alpha\varepsilon(k)e(k) - 2\beta\varepsilon(k)\phi(k) + \beta\varepsilon^2(k)] \geq -\gamma \quad (18)$$

Using the Schwartz inequality

$$\begin{aligned}\eta\{\|(\phi(k), e(k))\|_2^2\}_N - |\alpha| \cdot \{\|\varepsilon(k)\|_2\}_N \cdot \{\|e(k)\|_2\}_N - 2|\beta| \cdot \{\|\varepsilon(k)\|_2\}_N \cdot \{\|\phi(k)\|_2\}_N \\ \leq \gamma + |\beta| \cdot \{\|\varepsilon(k)\|_2^2\}_N\end{aligned} \quad (19)$$

Assume $\{\|(\phi(k), e(k))\|_2^2\}_N \rightarrow \infty$as $N \rightarrow \infty$, then from equation (19) we derive $\eta \leq 0$. This is a contradiction. Therefore $\{\|(\phi(k), e(k))\|_2^2\}_N$is bounded for all $N \in Z_+$, i.e., $\phi(k)$, $e(k) \in L_2$. ■

Theorem 2 *(Cluett's Law{2) For the feedback system (16), if*

i) $H_1 : e_0(k) - e(k)$ satisfies

$$\sum_{k=1}^{N} \left(e_0(k)e(k) + \sigma e_0(k)^2/2\right) \geq -\gamma$$

ii) $H_2 : e(k) - \phi(k)$ satisfies

$$\sum_{k=1}^{N} \left(\sigma\phi(k)^2/2 - \phi(k)e(k)\right) \leq -\eta\|(\phi(k), e(k))\|_{2,N}^2$$

for some $\gamma \geq 0$, $\eta > 0$, which are independent of N, and $\sigma \in (0, 1]$, which is independent of k and N, then the closed loop signals $e(k)$, $\phi(k) \in L_2$.

Proof: See corollary 2.1 in [15]. ■

Remark 2 *As a matter of fact, the operator H_1 represents the nonlinear mapping and H_2 is a dynamic linear transfer function. When condition (i) and (ii) are satisfied, H_2 is guaranteed to be passive and H_1^{-1} is strictly interior conic (c_1, r_1), where $c_1 = 1$ and $r_1 = (1- \sigma)^{1/2}$, or equivalently H_1 is strictly interior the conic (c_2, r_2) where $c_2 = \sigma^{-1}$ and $r_1 = \sigma^{-1} (1 - \sigma)^{1/2}$ as long as $\sigma < 1$ holds. Hence the feedback loop is L_2-stable by the conic sector theorem. This conic relation is illustrated in Figure 2*

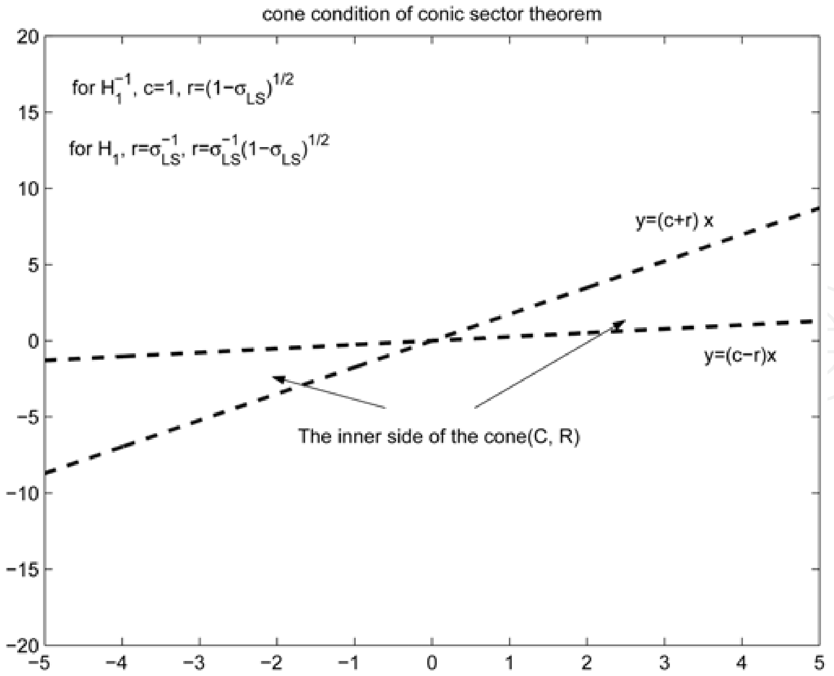


Figure 2. Illustration of interior and exterior conic relations of H_1

3.2 Output layer analysis of SISO RNN

In this and next section, we consider the RNN model of (1) with only one output node, i.e., $l = 1$. Such simplification is favorable for us to put more concentration on the basic ideas of the proof rather than the pure mathematics. Moreover, the results for SISO RNN will also be extended to the more general case of MIMO RNN in later sections. On the other hand, in a multi-layered RNN, it may not be able to update all the estimated weights within a single gradient approximation function. Hence we shall partition the training into different layers. Now with the assumption of SISO RNN, the training for output layer can be re-written as

$$\hat{V}(k+1) = \hat{V}(k) + \frac{\alpha^v(k)}{\rho^v(k)} e(k) \left(\Phi(k)^T + \beta^v(k)\hat{V}(k) diag\{\Phi'(k)\}\hat{W}(k)\hat{D}^v(k) \right) \quad (20)$$

In order to analyze the dynamics of this training equation via input-output approach, the first step is to restructure (20) into an error feedback loop, which should be the same as that in Figure 1. Further, the weight estimation error must be referred as the output signal. For this purpose, define the estimation error

$$e^v(k) = \hat{V}(k)\Phi(k) - V^*\Phi(k) = \tilde{V}(k)\Phi(k) \quad (21)$$

where $V^* \in R^{1\times m}$ and $\tilde{V}(k) = V(k) - V^*$ are the ideal weight vector and estimation error vector of output layer respectively, and $\Phi^*(k)$ is defined in analogous to $\Phi(k)$ as

$$\Phi^*(k) = [\phi(W_1^* x^*(k)) \quad \phi(W_2^* x^*(k)) \quad \cdots \quad \phi(W_m^* x^*(k))]^T \quad (22)$$

where $x^*(k) \in R^{n\times 1}$ is the ideal input state, $W^* \in R^{m\times n}$ is the ideal weight matrix of hidden layer of the RNN. Then the training error of RNN can be expanded as

$$\begin{aligned} e(k) &= d(k) - \hat{y}(k) + \varepsilon(k) \\ &= V^*\Phi^*(k) - \hat{V}(k)\Phi(k) + \varepsilon(k) \\ &= [V^*\Phi^*(k) - V^*\Phi(k)] - [\hat{V}(k)\Phi(k) - V^*\Phi(k)] + \varepsilon(k) \end{aligned} \tag{23}$$

Because the term V^* $\Phi^*(k)$ - V^* Φ (k) is temporarily constant in case of output layer training, we can define $\tilde{\varepsilon}^{v}(k)$ = ε (k) + V^* $\Phi^*(k)$ - V^* Φ (k). Then (23) can be transformed as

$$\tilde{\varepsilon}^v(k) - e^v(k) = e(k) \tag{24}$$

Equation (24) has a similar form as the feedback path of the system (16), with $e^v(k)$ and $e(k)$ corresponding to $e(k)$ and $e_0(k)$ in Figure 1 respectively, and here the feedback gain is unity, i.e., $H_2 = 1$.

There is an important implication in the relation of (24). The $e^v(k)$, $e(k)$ and $\tilde{\varepsilon}^{v}(k)$ correspond to the weight estimation error, the RNN modeling error and the disturbance, respectively. Hence the training error is directly linked to the disturbance, and in turn, the parameter estimating error of the RNN output layer. If we further establish a nonlinear mapping from the original disturbance $\tilde{\varepsilon}^{v}(k)$ to the parameter estimation error $e^v(k)$, the relationship between L_2-stability of training algorithm and learning parameters can subsequently be studied by imposing the conditions of Theorem 2.

Theorem 3 *If the output layer of the RNN is trained by the adaptive normalized gradient algorithm (20), the weight* $\hat{V}$ *(k) is guaranteed to be stable in the sense of Lyapunov*

$$\|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2 \le 0, \quad \forall k \tag{25}$$

with $\tilde{V}$ (k) = V (k) - V^*. Also the training will be L_2-stable in the sense of $e^v(k) \in L_2$ if $\alpha^v(k) \neq 0$ for all $k \in Z_+$.

Proof: Subtracting $V^*$and then squaring both sides of (20)

$$\begin{aligned} &\|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2 \\ = \ &\frac{2\alpha^v(k)e(k)}{\rho^v(k)} \cdot \tilde{V}(k)(\Phi(k)^T + \beta^v(k)\hat{A}(k))^T + (\frac{\alpha^v(k)e(k)}{\rho^v(k)})^2\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \\ = \ &\frac{2\alpha^v(k)e(k)}{\rho^v(k)} \cdot \tilde{V}(k)(\Phi(k) + \beta^v(k)\hat{A}(k)^T) + (\frac{\alpha^v(k)e(k)}{\rho^v(k)})^2\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \end{aligned} \tag{26}$$

Regarding the first term on the right side of (26), we find that it may be easily associated with the term $e^v(k)$ due to the explicit appearance of $\tilde{V}$ (k) and $\Phi(k)$. Following this idea, we need to apply certain transformation to $\beta^v(k)$ $\hat{A}$ $(k)^T$, such that $\Phi(k)$ can be extracted from the summation. When it comes to this point, our first thought is to left multiply $\beta^v(k)\hat{A}(k)^T$ by $\Phi(k)\Phi(k)^T(\Phi(k)\Phi(k)^T)^{-1}$. However, the transformation is not valid because $\Phi(k)$ $\Phi(k)^T$ is not an invertible matrix ($\Phi(k)$ is a column vector). Fortunately, inspired by the approximation method of classical Gauss-Newton iteration algorithm [2] (pp.126-127), we can add the term $\Phi(k)$ $\Phi(k)^T$ by a small positive constant δ to expand it into

$$\delta I + \Phi(k)\Phi(k)^T : \text{ positive definite for all } k \tag{27}$$

Such that the singular matrix problem can be avoided. On this basis, we have the following derivations

$$\begin{aligned}
& \|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2 \\
= \; & \frac{2\alpha^v(k)e(k)}{\rho^v(k)} \cdot \tilde{V}(k)\Phi(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
& +(\frac{\alpha^v(k)e(k)}{\rho^v(k)})^2\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \\
= \; & \frac{2\alpha^v(k)e(k)}{\rho^v(k)} e^v(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
& +(\frac{\alpha^v(k)e(k)}{\rho^v(k)})^2\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2
\end{aligned} \tag{28}$$

$$\begin{aligned}
= \; & \frac{2\alpha^v(k)(\tilde{\varepsilon}^v(k)e(k) - e^2(k))}{\rho^v(k)}(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
& +(\frac{\alpha^v(k)e(k)}{\rho^v(k)})^2\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2
\end{aligned} \tag{29}$$

where (29) is obtained by substituting (24) into (28). Then based on the triangular inequality $2\tilde{\varepsilon}^v(k)e(k) \leq (\tilde{\varepsilon}^v(k))^2 + e^2(k)$, (29) can be further deducted as

$$\begin{aligned}
& \|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2 \\
\leq \; & \frac{\alpha^v(k)((\tilde{\varepsilon}^v(k))^2 - e^2(k))}{\rho^v(k)}(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
& +(\frac{\alpha^v(k)}{\rho^v(k)})^2 e^2(k)\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \\
= \; & \frac{\alpha^v(k)}{\rho^v(k)}(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)\left((\tilde{\varepsilon}^v(k))^2\right. \\
& \left. -(1 - \frac{\alpha^v(k)\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)})\; e^2(k)\right)
\end{aligned}$$

By the definition of $\beta^v(k)$, we may derive that $\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T \geq 0$. Furthermore, because that $\rho^v(k) \geq \|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2$ as defined in (11) which lead to 1 - $\frac{\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)}$ > 0, and by the definition of $\alpha^v(k)$, the convergence of $\tilde{V}$ (k) can be derived

$$\begin{aligned}
&\|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2 \\
\leq\ & \frac{\alpha^v(k)}{\rho^v(k)}(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)\left((\tilde{\varepsilon}^v(k))^2\right. \\
& \left. -(1-\frac{\alpha^v(k)\|\Phi(k)^T+\beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)})\ e^2(k)\right) \\
\leq\ & \frac{\alpha^v(k)}{\rho^v(k)}(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)\left((\varepsilon^v_{max})^2\right. \\
& \left. -(1-\frac{\alpha^v(k)\|\Phi(k)^T+\beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)})\ e^2(k)\right) \\
\leq\ & 0
\end{aligned} \tag{30}$$

Next considering the case that the assumption $\alpha^v(k) \neq 0$ holds for all $k \in Z_+$, we can divide both sides of (28) by $2\alpha^v(k)(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)$ and then sum up to N steps

$$\begin{aligned}
-\Delta V &= \sum_{k=1}^{N}\left(\frac{e(k)e^v(k)}{\rho^v(k)} + \frac{\alpha^v(k)\|\Phi(k)^T+\beta^v(k)\hat{A}(k)\|^2}{2(\rho^v(k))^2(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)}e^2(k)\right) \\
&\leq \sum_{k=1}^{N}(\bar{e}(k)\bar{e}^v(k) + \frac{1}{2}\bar{\sigma}^v(\bar{e}(k))^2) \qquad \forall k \in \{k|\ \alpha^v(k) \neq 0\}
\end{aligned} \tag{31}$$

where the normalized error signals are defined as

$$\bar{e}(k) = \frac{e(k)}{\sqrt{\rho^v(k)}}, \quad \bar{e}^v(k) = \frac{e^v(k)}{\sqrt{\rho^v(k)}}$$

and the cone satisfies

$$\bar{\sigma}^v = \sup_k\{\frac{\|\Phi(k)^T+\beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)}\} < 1$$

which prevents the vanishing radius problem, i.e., $\bar{\sigma}^v$ is strictly smaller than one [15]. Because for each k the Lyapunov function (30) is guaranteed smaller or equal to zero, we have

$$\begin{aligned}
0 \quad \leq \quad \Delta V &= -\sum_{k=1}^{N}\frac{\|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2}{2(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)} \\
&\leq -\frac{1}{2}\sum_{k=1}^{N}(\|\tilde{V}(k+1)\|^2 - \|\tilde{V}(k)\|^2) \\
&= \frac{1}{2}(\|\tilde{V}(1)\|^2 - \|\tilde{V}(N+1)\|^2)
\end{aligned}$$

Due to the specific selection of the normalization factor in (11), the normalized error signals guarantee that the original signals $e(k)$ and $e^v(k)$ are bounded according to the original operators H_1^v and H_2 [15]. Now the operator H_1^v represented by (31) satisfies the condition (i) of Theorem 2, and condition (ii) is guaranteed to hold due to $H_2 = 1$. Thus we conclude that $e^v(k) \in L_2$. ■

Remark 3 *According to the theoretical analysis, the three adaptive parameters $\alpha^v(k)$, $\beta^v(k)$ and $\rho^v(k)$ play important roles in the design of the RAGD. The adaptive learning rate $\alpha^v(k)$ is based on the standard adaptive control system to solve the weight drift problem [10]. The normalization factor $\rho^v(k)$ prevents the so-called vanishing cone problem of the conic sector theorem [15], which also has a similar role to the local stability condition as in [8] to bound the gradient in (20). The specific designed hybrid adaptive learning rate $\beta^v(k)$ can be further interpreted as activating the recurrent learning fashion in case $\Phi(k)^T\{\delta I + \Phi(k)\Phi(k)^T\}^{-1}\hat{A}(k)^T \geq 0$. It implies that the recurrent training of the RAGD will be active only if the second term of the derivative in (20) gives the negative gradient direction, i.e., a relatively deeper local attractor, otherwise the RAGD training procedure will be the same as a static BP algorithm and likely escape this undesired local attractor since it is unfavorable in the recurrent training. This design is especially effective for accelerating the training of the RNN when the iteration is near the bottom of basin of a local attractor, where the derivatives are changed slowly. With $\beta^v(k) = 1$, the approximation of $\hat{D}^v(k)$ is more accurate to meet the convergence and stability requirements.*

Remark 4 *The idea of the RAGD is similar to the existing works [16] [17] [14]. If we calculate the derivative in (20) exactly by unfolding the recurrent structure and force $\beta^v(k) = 0$, i.e, pursuing all N steps back in the past, then the algorithm will recover the static BP [17] [18]. Moreover, based on the assumption that the model parameters do not change apparently between each iteration [16], then we can derive a similar approach as the N-RTRL [14]. However, the key difference between the RAGD and the N-RTRL is that we use the hybrid learning rate $\beta^v(k)$ to guarantee the weight convergence and system stability.*

3.3 Hidden layer analysis of SISO RNN

This section presents the stability analysis for the hidden layer training of the RAGD. Apparently the analysis for the hidden layer is more di±cult than the one of the output layer, because the dynamics between the weight and modeling error is nonlinear. The derivation of error gradient must be carried out through one layer backward, which involves the derivative of activation function. In the following analysis, we show that the nonlinearity can actually be avoided by using the mean value theorem. On the other hand, as mentioned in section 2, the Frobenius norm is employed as weight matrix norm in the proof, e.g., $\left\|\hat{W}(k)\right\|_F$. A direct benefit of this expression is that the proof and the training equation can be presented in matrix forms, while not in a manner of row by row. However question arises, it is difficult to derive the Jacobian in this framework. We find that it is feasible to extend the Jacobian into a long vector form on the row basis. Next, similar to the output layer analysis, the hidden layer training of the RAGD of SISO RNN can be simplified as follows

$$\hat{W}(k+1) = \hat{W}(k) + \frac{\alpha^w(k)}{\rho^w(k)} \cdot e(k)diag\{\Phi^{'}(k)\}\hat{V}(k)^T\left(\hat{x}(k)^T + \beta^w(k)\underline{\hat{W}}(k)\hat{D}^w(k)\right) \quad (32)$$

Expanding the modeling error around the hidden layer weight

$$\begin{aligned}
e(k) &= d(k) - \hat{y}(k) + \varepsilon(k) \\
&= V^*\Phi^*(k) - \hat{V}(k)\Phi(k) + \varepsilon(k) \\
&= V^*\Phi^*(k) - \hat{V}(k)\Phi(W^*\hat{x}(k)) + \hat{V}(k)\Phi(W^*\hat{x}(k)) - \hat{V}(k)\Phi(\hat{W}(k)\hat{x}(k)) + \varepsilon(k) \\
&= \hat{V}(k)\Phi(W^*\hat{x}(k)) - \hat{V}(k)\Phi(\hat{W}(k)\hat{x}(k)) + \tilde{\varepsilon}^w(k) \\
&= -\hat{V}_1(k)\mu_1(k)\tilde{W}_1(k)\hat{x}(k) - \hat{V}_2(k)\mu_2(k)\tilde{W}_2(k)\hat{x}(k) \cdots - \hat{V}_m(k)\mu_m(k)\tilde{W}_m(k)\hat{x}(k) + \tilde{\varepsilon}^w(k) \\
&= -\sum_{i=1}^{m} \hat{V}_i(k)\mu_i(k)\tilde{W}_i(k)\hat{x}(k) + \tilde{\varepsilon}^w(k) \\
&= -\hat{V}(k)diag\{\Psi(k)\}\tilde{W}(k)\hat{x}(k) + \tilde{\varepsilon}^w(k) \qquad (33)
\end{aligned}$$

where $\tilde{\varepsilon}^w(k) = V^*\Phi^*(k) - \hat{V}(k)\Phi(W^*\hat{x}(k)) + \varepsilon(k)$, $\tilde{W}_i(k) \in R^{1\times n}$is the vector difference between the *i*th row of \hat W(k) and the ideal weight *W**, $\mu_i(k)$ is the mean value of the *i*th nonlinear activation function, and $\Psi(k)$ is

$$\Psi(k) = [\mu_1(k), \mu_2(k), \cdots \mu_m(k)]^T$$

Defining

$$e^w(k) = \hat{V}(k)diag\{\Psi(k)\}\tilde{W}(k)\hat{x}(k) \qquad (34)$$

then equation (33) can be simplified as

$$e(k) = -e^w(k) + \tilde{\varepsilon}^w(k) \qquad (35)$$

Because the output layer weight is always updated before the hidden layer weight, and $\hat{V}(k)$ of the RAGD is bounded as already proved in Section 3.2, then definitely the error signal $\tilde{\varepsilon}^w(k)$ is also bounded for every step *k*. Furthermore, since $H_2 = 1$ is inside any cone, thus we only need to study the operator H_1 to analyze the stability of the training.

Theorem 4 *If the output layer of the RNN is trained by the adaptive normalized gradient algorithm (32), the weight matrix* $\hat{W}(k)$ *is guaranteed to be stable in the sense of Lyapunov*

$$\|\tilde{W}(k+1)\|_F^2 - \|\tilde{W}(k)\|_F^2 \le 0, \qquad \forall k$$

with $\tilde{W}(k) = \hat{W}(k) - W^*$. *Also the hidden layer training of the RAGD will be* L_2*-stable in the sense of* $e^w(k) \in L_2$ *if* $\alpha^w(k) \neq 0$ *for all* $k \in Z_+$.

Proof: Subtracting *W** from both sides of (32)

$$\tilde{W}(k+1) = \tilde{W}(k) + \frac{\alpha^w(k)}{\rho^w(k)} \cdot e(k)\frac{d\hat{y}(k)}{d\hat{W}(k)} \qquad (36)$$

Squaring both sides of (36)

$$
\begin{aligned}
& \tilde{W}(k+1)^T\tilde{W}(k+1) \\
= \ & (\tilde{W}(k) + \frac{\alpha^w(k)}{\rho^w(k)} \cdot e(k)\frac{d\hat{y}(k)}{d\hat{W}(k)})^T(\tilde{W}(k) + \frac{\alpha^w(k)}{\rho^w(k)} \cdot e(k)\frac{d\hat{y}(k)}{d\hat{W}(k)}) \\
= \ & \tilde{W}(k)^T\tilde{W}(k) + \frac{\alpha^w(k)e(k)}{\rho^w(k)}\tilde{W}(k)^T \cdot \frac{d\hat{y}(k)}{d\hat{W}(k)} + \frac{\alpha^w(k)e(k)}{\rho^w(k)}\frac{d\hat{y}(k)}{d\hat{W}(k)^T} \cdot \tilde{W}(k) \\
& + \frac{(\alpha^w(k))^2e^2(k)}{(\rho^w(k))^2}\frac{d\hat{y}(k)}{d\hat{W}(k)^T} \cdot \frac{d\hat{y}(k)}{d\hat{W}(k)}
\end{aligned}
\tag{37}
$$

By the definition of Frobenius norm

$$
\begin{cases}
Trace\{\tilde{W}(k+1)^T\tilde{W}(k+1)\} = \|\tilde{W}(k+1)\|_F^2 \\
Trace\{\tilde{W}(k)^T\tilde{W}(k)\} = \|\tilde{W}(k)\|_F^2 \\
Trace\{\frac{d\hat{y}(k)}{d\hat{W}^T(k)} \cdot \frac{d\hat{y}(k)}{d\hat{W}(k)}\} = \left\|\frac{d\hat{y}(k)}{d\hat{W}(k)}\right\|_F^2 = \|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T + \beta^w(k)\hat{B}(k))\|_F^2 \\
Trace\{\tilde{W}(k)^T \cdot \frac{d\hat{y}(k)}{d\hat{W}(k)}\} = Trace\{\frac{d\hat{y}(k)}{d\hat{W}(k)^T} \cdot \tilde{W}(k)\}
\end{cases}
$$

where *Trace* {•} function is defined as the sum of the entries on the main diagonal of the associated matrix. The following equation can be derived then

$$
\begin{aligned}
\|\tilde{W}(k+1)\|_F^2 - \|\tilde{W}(k)\|_F^2 = \frac{2\alpha^w(k)e(k)}{\rho^w(k)}Trace\{\frac{d\hat{y}(k)}{d\hat{W}(k)^T}\tilde{W}(k)\} \\
+ \frac{(\alpha^w(k))^2e^2(k)}{(\rho^w(k))^2} \cdot \|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T + \beta^w(k)\hat{B}(k))\|_F^2
\end{aligned}
\tag{38}
$$

Using the trace properties, the first term on the right side of (38) can be transformed as

$$
\begin{aligned}
& e(k)Trace\{\frac{d\hat{y}(k)}{d\hat{W}(k)^T}\tilde{W}(k)\} \\
= \ & e(k)Trace\{\left(\hat{x}(k) + \beta^w(k)\hat{B}(k)^T\right)\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\} \\
= \ & e(k)Trace\{\left(\hat{x}(k)\hat{V}(k)diag\{\Phi^{'}(k)\} + \beta^w(k)\hat{B}(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\right)\tilde{W}(k)\} \\
= \ & e(k)Trace\{\hat{x}(k)\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\} + e(k)\beta^w(k)Trace\{\hat{B}(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\} \\
= \ & e(k)Trace\{\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)\} + e(k)\beta^w(k)Trace\{\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{B}(k)^T\} \\
= \ & e(k)\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k) + e(k)\beta^w(k)V(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k) \\
= \ & e(k)\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)(1 + \beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \\
= \ & e(k)(\sum_{i=1}^{m}\hat{V}_i(k)\Phi_i^{'}(k)\tilde{W}_i(k)\hat{x}(k))(1 + \beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)
\end{aligned}
\tag{39}
$$

where the third equality to the last is derived by the similar perturbation method as the one in the output layer training (adding a small constant diagonal matrix δ*I* to $\hat{x}(k)\hat{x}(k)^T$to make it invertible, see the proof in Section 3.2).

Before proceeding, let's consider a RNN with scalar weight $\hat{W}$ (*k*). The relation of the local attractor basin of the instantaneous square error against the $\tilde{W}$ (*k*) can be presented by $-\frac{df(e(k))}{\hat{W}(k)}\tilde{W}(k)$, as illustrated in Figure 3 [10]. Extend this result to the RNN with a matrix weight $\hat{W}$ (*k*), we have a similar presentation by the local attractor basin concept

$$-\frac{df(e(k))}{\hat{W}_i(k)^T}\tilde{W}_i(k) \leq 0, \qquad \forall k, i \tag{40}$$

By the local attractor basin properties in (40)

$$\begin{aligned} & e(k)(\hat{V}_i(k)\Phi_i^{'}(k)\tilde{W}_i(k)\hat{x}(k))(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \\ & = e(k)\frac{d\hat{y}(k)}{\hat{W}_i(k)^T}\tilde{W}_i(k) = -\frac{df(e(k))}{\hat{W}_i(k)^T}\tilde{W}_i(k) \leq 0 \end{aligned} \tag{41}$$

The right side of (39) can be enlarged as

$$\begin{aligned} & e(k)(\sum_{i=1}^{m}\frac{\Phi_i^{'}(k)}{\mu_i(k)}\cdot\hat{V}_i(k)\mu_i(k)\tilde{W}_i(k)\hat{x}(k))(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \\ \leq \quad & e(k)(\frac{\phi_{min}^{\prime}(k)}{\mu_{max}})\cdot(\sum_{i=1}^{m}\hat{V}_i(k)\mu_i(k)\tilde{W}_i(k)\hat{x}(k))(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \\ = \quad & \frac{\phi_{min}^{\prime}(k)}{\mu_{max}}\cdot e(k)e^w(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \end{aligned} \tag{42}$$

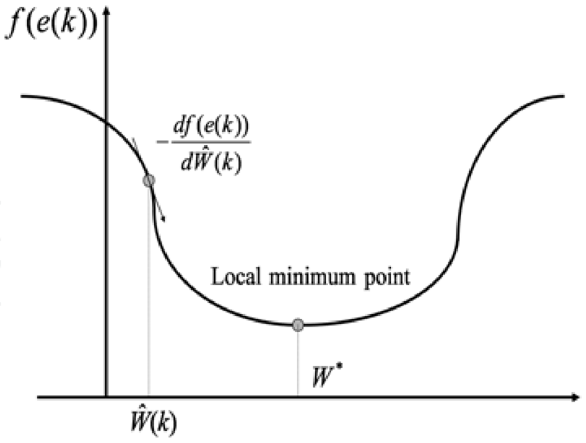


Figure 3. Illustration of a local attractor basin of the RNN against a scalar estimated weight $\hat{W}$ (*k*)

Substituting (42) into (39)

$$\begin{aligned} & \|\tilde{W}(k+1)\|_F^2 - \|\tilde{W}(k)\|_F^2 \\ \leq \quad & \frac{2\alpha^w(k)\phi_{min}^{\prime}(k)e(k)e^w(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \end{aligned}$$

$$+\frac{(\alpha^w(k))^2 e^2(k)}{(\rho^w(k))^2}\cdot \|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2 \tag{43}$$

Substituting (35) into (43)

$$\begin{aligned}
&\|\tilde{W}(k+1)\|_F^2-\|\tilde{W}(k)\|_F^2\\
\leq\ & \frac{2\alpha^w(k)\phi'_{min}(k)(\bar{\varepsilon}^w(k)e(k)-e^2(k))}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)\\
&+\frac{(\alpha^w(k))^2e^2(k)}{(\rho^w(k))^2}\cdot\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2\\
\leq\ & \frac{\alpha^w(k)\phi'_{min}(k)((\bar{\varepsilon}^w(k))^2-e^2(k))}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)\\
&+\frac{(\alpha^w(k))^2e^2(k)}{(\rho^w(k))^2}\cdot\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2\\
=\ & \frac{\alpha^w(k)\phi'_{min}(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)((\bar{\varepsilon}^w(k))^2\\
&-(1-\frac{\alpha^w(k)\mu_{max}\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2}{\rho^w(k)\phi'_{min}(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)})e^2(k))\\
\leq\ & \frac{\alpha^w(k)\phi'_{min}(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)((\varepsilon^w_{max})^2\\
&-(1-\frac{\alpha^w(k)\mu_{max}\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2}{\rho^w(k)\phi'_{min}(k)})e^2(k))
\end{aligned}\tag{44}$$

By the definition of $\rho^w(k)$ and $\alpha^w(k)$ in (12) and (14) respectively, we can draw that

$$\|\tilde{W}(k+1)\|_F^2-\|\tilde{W}(k)\|_F^2\leq 0 \tag{45}$$

Again, consider the extreme case with the assumption of nonzero $\alpha^w(k)$. Dividing both sides of (43) by

$$\frac{2\alpha^w(k)\phi'_{min}(k)}{\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)$$

and then summing up to N steps

$$\begin{aligned}
-\Delta W &\leq \sum_{k=1}^{N}\left(\frac{e(k)e^w(k)}{\rho^w(k)}+\frac{\alpha^w(k)\mu_{max}\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2}{2(\rho^w(k))^2\phi'_{min}(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)}e^2(k)\right)\\
&\leq \sum_{k=1}^{N}\{\bar{e}(k)\bar{e}^w(k)+\tfrac{1}{2}\bar{\sigma}^w\bar{e}^2(k)\}
\end{aligned}\tag{46}$$

where the normalized error signals are $\bar{e}(k)=e(k)/\sqrt{\rho^w(k)}$, $\bar{e}^w(k)=e^w(k)/\sqrt{\rho^w(k)}$, and the cone is

$$\bar{\sigma}^w = \sup_k\{\frac{\mu_{max}\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T(\hat{x}(k)^T + \beta^w(k)\hat{B}(k))\|_F^2}{\rho^w(k)\phi'_{min}(k)}\} < 1 \tag{47}$$

and Δ*W* is greater than zero because for each *k* the Lyapunov function (45) is guaranteed smaller than or equal to zero, i.e.

$$\begin{aligned} 0 \quad \leq \quad \Delta W &= -\sum_{k=1}^{N}\frac{\mu_{max}(\|\tilde{W}(k+1)\|_F^2 - \|\tilde{W}(k)\|_F^2)}{2\phi'_{min}(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)} \\ \leq \quad & \frac{\mu_{max}}{2min\{\phi'_{min}(k)\}}\sum_{k=1}^{N}(\|\tilde{W}(k)\|_F^2 - \|\tilde{W}(k+1)\|_F^2) \\ = \quad & \frac{\mu_{max}}{2min\{\phi'_{min}(k)\}}(\|\tilde{W}(1)\|_F^2 - \|\tilde{W}(N+1)\|_F^2) \end{aligned} \tag{48}$$

Due to the specific selection of the normalization factor in (12), the original signals *e*(*k*) and $e^w(k)$ are guaranteed to be bounded according to the original operators H_1^w and H_2 [11] [15]. Now the operator H_1^w represented by (46) satisfies the condition (i) of Theorem 2. Thus we conclude that $e^w(k) \in L_2$ in case of $\alpha^w(k) \neq 0$, $\forall k \in Z_+$. ■

3.4 Robustness analysis of MIMO RNN

In this section, we discuss the RAGD training for the RNN of Multi-Input Multi-Output (MIMO) types. As mentioned in the introduction, the RNN with multiple output neurons can be regarded as consisting of several single output RNNs. Thus the training of MIMO RNN can be studied by decomposition. In detail, for the output layer training, we may calculate the gradient of each output neuron with respect to weight parameters, and then obtain the total weight updating by summing these individual gradient. As for the hidden layer, we also use this method to take into account the influence of multi-output neurons on total weight updating. Following this idea, the extension of the stability analysis from SISO to MIMO is straight forward.

Theorem 5 *If the RNN is trained by the adaptive normalized gradient algorithm (5)-(15), then the weight* $\hat{V}$ *(k) and* $\hat{W}$ *(k) are guaranteed to be stable in the sense of Lyapunov.*

Proof: *(i) Output layer analysis:* To study the stability of the RAGD, we need to establish the error dynamics of the training algorithm. First of all, define the estimation error

$$e^v(k) = \hat{V}(k)\Phi(k) - V^*\Phi(k) = \tilde{V}(k)\Phi(k) \tag{49}$$

where $V^* \in R^{l\times m}$ is the ideal output layer weight, and

$$\begin{cases} \tilde{V}(k) = V(k) - V^* \\ \Phi^*(k) = [\quad\cdots\quad h(W_i^*(k)x^*(k)) \quad\cdots\quad]^T \end{cases}$$

Then we expand $e(k) \in R^{l\times 1}$ with respect to the output layer weight as

$$\begin{aligned} e(k) &= d(k) - \hat{y}(k) + \varepsilon(k) \\ &= [V^*\Phi^*(k) - V^*\Phi(k)] - [\hat{V}(k)\Phi(k) - V^*\Phi(k)] + \varepsilon(k) \\ &= \tilde{\varepsilon}^v(k) - e^v(k) \end{aligned} \tag{50}$$

with $\tilde{\varepsilon}^{v}(k) = V^*\Phi^*(k) - V^*\Phi(k) + \varepsilon(k)$. In (50), we restructure the output layer training of the RAGD algorithm into a closed loop form same as that of (16) , by which the weight estimation error $e^v(k)$ is referred as the output signal. Subtracting V^* and squaring both sides of the output layer training equation in (5)

$$\begin{aligned} \|\tilde{V}(k+1)\|_F^2 &= \|\tilde{V}(k)\|_F^2 + \frac{2\alpha^v(k)}{\rho^v(k)} Trace\{(\Phi(k)^T + \beta^v(k)\hat{A}(k))^T e(k)^T \tilde{V}(k)\} \\ &\quad + (\frac{\alpha^v(k)}{\rho^v(k)})^2 \|e(k)\Phi(k)^T + e(k)\beta^v(k)\hat{A}(k)\|_F^2 \end{aligned} \tag{51}$$

By the matrix trace properties

$$\begin{aligned} & Trace\{(\Phi(k)^T + \beta^v(k)\hat{A}(k))^T e(k)^T \tilde{V}(k)\} \\ = \; & Trace\{\Phi(k)e(k)^T\tilde{V}(k)\} + \beta^v(k) Trace\{\hat{A}(k)^T e(k)^T \tilde{V}(k)\} \\ = \; & Trace\{e(k)^T\tilde{V}(k)\Phi(k)\} + \beta^v(k) Trace\{e(k)^T\tilde{V}(k)\hat{A}(k)^T\} \\ = \; & e(k)^T\tilde{V}(k)\Phi(k) + \beta^v(k)e(k)^T\tilde{V}(k)\hat{A}(k)^T \end{aligned}$$

Again, we employ the customary practice by using a small positive perturbation constant δ to make $\delta I + \Phi(k)\Phi(k)^T$ full rank and then apply the approximation as

$$\begin{aligned} & Trace\{(\Phi(k)^T + \beta^v(k)\hat{A}(k))^T e(k)^T \tilde{V}(k)\} \\ = \; & e(k)^T\tilde{V}(k)\Phi(k) + \beta^v(k)e(k)^T\tilde{V}(k)\Phi(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T \\ = \; & e(k)^T e^v(k) + \beta^v(k)e(k)^T e^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T \\ = \; & e(k)^T e^v(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \end{aligned} \tag{52}$$

Substituting (50) and (52) into (51)

$$\begin{aligned} & \|\tilde{V}(k+1)\|_F^2 - \|\tilde{V}(k)\|_F^2 \\ = \; & \frac{2\alpha^v(k)}{\rho^v(k)} e(k)^T e^v(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\ & + (\frac{\alpha^v(k)}{\rho^v(k)})^2 \|e(k)\Phi(k)^T + e(k)\beta^v(k)\hat{A}(k)\|_F^2 \\ \leq \; & \frac{2\alpha^v(k)}{\rho^v(k)} e(k)^T e^v(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\ & + (\frac{\alpha^v(k)}{\rho^v(k)})^2 \cdot \|e(k)\|^2 \cdot \|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \end{aligned} \tag{53}$$

$$\begin{aligned}
&= \frac{2\alpha^v(k)}{\rho^v(k)}(e(k)^T\tilde{\varepsilon}^v(k) - \|e(k)\|^2)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
&\quad +(\frac{\alpha^v(k)}{\rho^v(k)})^2 \cdot \|e(k)\|^2 \cdot \|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \\
&\leq \frac{\alpha^v(k)}{\rho^v(k)}(\|\tilde{\varepsilon}^v(k)\|^2 - \|e(k)\|^2)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
&\quad +(\frac{\alpha^v(k)}{\rho^v(k)})^2 \cdot \|e(k)\|^2 \cdot \|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2 \qquad (54)
\end{aligned}$$

$$\begin{aligned}
&= \frac{\alpha^v(k)}{\rho^v(k)}(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)(\|\tilde{\varepsilon}^v(k)\|^2 \\
&\quad -(1 - \frac{\alpha^v(k)\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)})\|e(k)\|^2) \\
&\leq \frac{\alpha^v(k)}{\rho^v(k)}(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)(\|\tilde{\varepsilon}^v(k)\|^2 \\
&\quad -(1 - \frac{\alpha^v(k)\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)})\|e(k)\|^2) \\
&\leq \frac{\alpha^v(k)}{\rho^v(k)}(1 + \beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)((\varepsilon^v_{max})^2 \\
&\quad -(1 - \frac{\alpha^v(k)\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)})\|e(k)\|^2) \qquad (55)
\end{aligned}$$

where (54) is because of the triangular inequality $\tilde{\varepsilon}^v(k)^T e(k) \leq (\|\tilde{\varepsilon}^v(k)\|^2 + \|e(k)\|^2)/2$, and (55) is due to

$$\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T \geq 0$$

Combining the inequality (55) with the definition of $\rho^{\ v}(k)$ and $\alpha^{\ v}(k)$ in (11) and (13) respectively, the Lyapunov equation of output layer estimation error can be derived

$$\|\tilde{V}(k+1)\|_F^2 - \|\tilde{V}(k)\|_F^2 \leq 0 \qquad (56)$$

(ii) Hidden layer analysis: Expanding $e(k)$ with respect to the estimation error of hidden layer weight

$$\begin{aligned}
e^w(k) &= \hat{V}(k)\Phi(\hat{W}(k)\hat{x}(k)) - \hat{V}(k)\Phi(W^*\hat{x}(k)) \\
&= \sum_{i=1}^{l}\sum_{j=1}^{m} e_i(k)\hat{V}_{i,j}(k)\mu_j(k)\tilde{W}_j(k)\hat{x}(k) \qquad (57)
\end{aligned}$$

where $W^*(k) \in R^{m\times n}$ is the ideal hidden layer weight matrices, $x^*(k) \in R^{n\times 1}$ is the ideal state vector, $\mu_i(k)$ is the mean value of the ith nonlinear activation function at instant k, and $\tilde{W}_{\ j}(k) = \hat{W}_{\ j}(k) \text{-} W_j^*$. Using the local attractor basin concept that similar to (40)

$$
\begin{aligned}
& Trace\{\left(\hat{x}(k)^T+\beta^w(k)\hat{B}(k)\right)^T e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\} \\
= \;& Trace\{\hat{x}(k)e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\}+\beta^w(k)Trace\{\hat{B}(k)^T e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\} \\
= \;& Trace\{e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)\}+\beta^w(k)Trace\{e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{B}(k)^T\} \\
= \;& e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)+\beta^w(k)e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{B}(k)^T \\
= \;& e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k) \\
& +\beta^w(k)e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T \\
= \;& e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\hat{x}(k)\left(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T\right) \\
= \;& (\sum_{i=1}^{l}\sum_{j=1}^{m}e_i(k)\hat{V}_{i,j}(k)\Phi_j^{'}(k)\tilde{W}_j(k)\hat{x}(k))\left(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T\right) \\
= \;& (\sum_{i=1}^{l}\sum_{j=1}^{m}\frac{\Phi_j^{'}(k)}{\mu_j(k)}e_i(k)\hat{V}_{i,j}(k)\mu_j(k)\tilde{W}_j(k)\hat{x}(k))\left(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T\right) \\
\leq \;& \frac{\phi_{min}^{'}(k)}{\mu_{max}}(\sum_{i=1}^{l}\sum_{j=1}^{m}e_i(k)\hat{V}_{i,j}(k)\mu_j(k)\tilde{W}_j(k)\hat{x}(k))(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \\
= \;& \frac{\phi_{min}^{'}(k)}{\mu_{max}}e^w(k)^T e(k)\left(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T\right)
\end{aligned}
\tag{58}
$$

Substituting *W**and squaring both sides of hidden layer training equation of the RAGD in (5), we can derive the Lyapunov function of the hidden layer weight of MIMO RNN based upon (58) as

$$
\begin{aligned}
& \|\tilde{W}(k+1)\|_F^2-\|\tilde{W}(k)\|_F^2 \\
= \;& \frac{2\alpha^w(k)}{\rho^w(k)}Trace\{(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))^T e(k)^T\hat{V}(k)diag\{\Phi^{'}(k)\}\tilde{W}(k)\} \\
& +(\frac{\alpha^w(k)}{\rho^w(k)})^2\|diag\{\Phi^{'}(k)\}\hat{V}(k)^T e(k)(\hat{x}(k)^T+\beta^w(k)\hat{B}(k))\|_F^2 \\
\leq \;& \frac{2\alpha^w(k)\phi_{min}^{'}(k)}{\rho^w(k)\mu_{max}}e^w(k)^T e(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T) \\
& +(\frac{\alpha^w(k)}{\rho^w(k)})^2\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2\cdot\|e(k)\|^2\cdot\|\hat{x}(k)^T+\beta^w(k)\hat{B}(k)\|^2
\end{aligned}
\tag{59}
$$

$$
\begin{aligned}
= \;& \frac{2\alpha^w(k)\phi_{min}^{'}(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)(\bar{\varepsilon}^w(k)^T e(k)-e(k)^T e(k)) \\
& +(\frac{\alpha^w(k)}{\rho^w(k)})^2\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2\cdot\|e(k)\|^2\cdot\|\hat{x}(k)^T+\beta^w(k)\hat{B}(k)\|^2 \\
\leq \;& \frac{\alpha^w(k)\phi_{min}^{'}(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I+\hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)(\|\bar{\varepsilon}^w(k)\|^2-\|e(k)\|^2)
\end{aligned}
$$

$$
\begin{aligned}
& +(\frac{\alpha^w(k)}{\rho^w(k)})^2 \|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|e(k)\|^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2 \\
= \;& \frac{\alpha^w(k)\phi'_{min}(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)(\|\bar{\varepsilon}^w(k)\|^2 \\
& -(1-\frac{\alpha^w(k)\mu_{max}\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2}{\rho^w(k)\phi'_{min}(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)})\|e(k)\|^2) \\
\leq \;& \frac{\alpha^w(k)\phi'_{min}(k)}{\rho^w(k)\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)((\varepsilon^w_{max})^2 \\
& -(1-\frac{\alpha^w(k)\mu_{max}\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2}{\rho^w(k)\phi'_{min}(k)})\|e(k)\|^2) \\
\leq \;& 0
\end{aligned}
\tag{60}
$$

Summarizing (56) and (60), we can conclude the proof. ■

Theorem 6 *If a MIMO RNN is trained by the adaptive normalized gradient algorithm (5)-(15), and $\alpha^v(k)$, $\alpha^w(k)$ are nonzero for all $k \in Z_+$, then the training will be L_2-stable in the sense of $e^v(k)$, $e^w(k) \in L_2$.*

Proof: Respectively, dividing both sides of (53) and (59) by the following two factors (since $\alpha^v(k)$; $\alpha^w(k) \neq 0$)

$$
\left\{
\begin{array}{l}
2\alpha^v(k)(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T) \\
\frac{2\alpha^w(k)\phi'_{min}(k)}{\mu_{max}}(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)
\end{array}
\right.
\tag{61}
$$

Summing both inequalities up to N steps, then for the output layer

$$
\begin{aligned}
-\Delta V_N = & \sum_{k=1}^{N}\{\frac{e^v(k)^T e(k)}{\rho^v(k)} + \frac{\alpha^v(k)\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{2(\rho^v(k))^2(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)}\|e(k)\|^2\} \\
& \leq \sum_{k=1}^{N}\{\bar{e}^v(k)^T\bar{e}_v(k) + \tfrac{1}{2}\bar{\sigma}^v\|\bar{e}_v(k)\|^2\}
\end{aligned}
\tag{62}
$$

and for the hidden layer

$$
-\Delta W_N \leq \sum_{k=1}^{N}\{\frac{e^w(k)^T e(k)}{\rho^w(k)} + \frac{\mu_{max}\alpha^w(k)\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2}{2(\rho^w(k))^2\phi'_{min}(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)}\|e(k)\|^2\}
$$

$$
\leq \sum_{k=1}^{N}\{\bar{e}^w(k)^T\bar{e}_w(k) + \tfrac{1}{2}\bar{\sigma}^w\|\bar{e}_w(k)\|^2\}
\tag{63}
$$

where

$$0 \leq \Delta V_N \leq \frac{1}{2}(\|\tilde{V}(1)\|_F^2 - \|\tilde{V}(N+1)\|_F^2)$$
$$0 \leq \Delta W_N \leq \frac{\mu_{max}}{2min\{\phi'_{min}(k)\}}(\|\tilde{W}(1)\|_F^2 - \|\tilde{W}(N+1)\|_F^2)$$

and the normalized signals are defined by

$$\bar{e}_v(k) = \frac{e(k)}{\sqrt{\rho^v(k)}} \qquad \bar{e}_w(k) = \frac{e(k)}{\sqrt{\rho^w(k)}}$$
$$\bar{e}^v(k) = \frac{e^v(k)}{\sqrt{\rho^v(k)}} \qquad \bar{e}^w(k) = \frac{e^w(k)}{\sqrt{\rho^w(k)}}$$

$$\bar{\sigma}^v(k) = \sup_k \{ \frac{\|\Phi(k)^T + \beta^v(k)\hat{A}(k)\|^2}{\rho^v(k)(1+\beta^v(k)\Phi(k)^T(\delta I + \Phi(k)\Phi(k)^T)^{-1}\hat{A}(k)^T)} \}$$
$$\bar{\sigma}^w(k) = \sup_k \{ \frac{\mu_{max}\|diag\{\Phi'(k)\}\hat{V}(k)^T\|_F^2 \cdot \|\hat{x}(k)^T + \beta^w(k)\hat{B}(k)\|^2}{\rho^w(k)\phi'_{min}(k)(1+\beta^w(k)\hat{x}(k)^T(\delta I + \hat{x}(k)\hat{x}(k)^T)^{-1}\hat{B}(k)^T)} \}$$

Due to the specific selection of the normalization factor $\rho^v(k)$ and $\rho^w(k)$ as in (11) and (12), the normalized error signals $\overline{e}_v(k)$, $\overline{e}^v(k)$, $\overline{e}_w(k)$, and $\overline{e}^w(k)$ are guaranteed to be bounded. Now, for each $\hat{V}$ (k) and $\overline{W}$ (k), applying the Cluett's law, we found that the operator H_1^v and H_1^w represented by (62) and (63) satisfy the condition (i). Further, $H_2 = 1$ ensures condition (ii) holds, thus $e^v(k)$and $e^w(k)$ are L_2 stable with $\alpha^v(k)$, $\alpha^w(k) \neq 0$, $\forall k \in Z_+$. ■

3.5 Summary

In Section 3, we introduce a novel RAGD training algorithm of RNN. Because conventional gradient type algorithms most likely suffer from slow convergence when dealing with statistically non-stationary inputs, the RAGD aims at overcoming such shortcomings via a series of new training parameters. Moreover, the robust local stability of the RAGD has been addressed for three layer RNN based upon the Cluett's law. Theoretical analysis shows that the proposed adaptive parameters improve the training performance in terms of a deeper gradient descent direction updating, which leads to a better transient response. Further, compared to BPTT, the RAGD algorithm requires limited backward unfolding, which reduces the computational complexity. The flow chart of the overall training procedure of the RAGD is summarized in Figure 4.

4. Applications in realtime signal processing

This section presents quantitative studies of the RAGD algorithm via computer simulations. We choose three of the most representative applications of RNN to verify the effectiveness of the RAGD. By default, the RNN is constructed with 50 hidden neurons and 5 input nodes. The 5- dimensional input vector consists of current and last sample of time sequence $u(k)$ and RNN output feedback with 1 to 3 steps delay respectively. Both hidden and output layer weights are initialized as uniformly distributed in the interval of (-1, 1). Sigmoid function is chosen as activation function, which is monotonic increasing, and both first and second order differentiable. The function and its first order derivative are given in equation (64), including the boundaries

$$\begin{cases} 0 \le \phi(x) = \frac{1}{1+e^{-\lambda x}} \le 1 \\ 0 \le \phi'(x) = \frac{\lambda e^{-\lambda x}}{(1+e^{-\lambda x})^2} \le \frac{\lambda}{4} \end{cases} \tag{64}$$

For the purpose of comparison, in most of the simulations we also provide the results of other training algorithms, such as the Truncated BPTT (T-BPTT) and the N-RTRL etc.

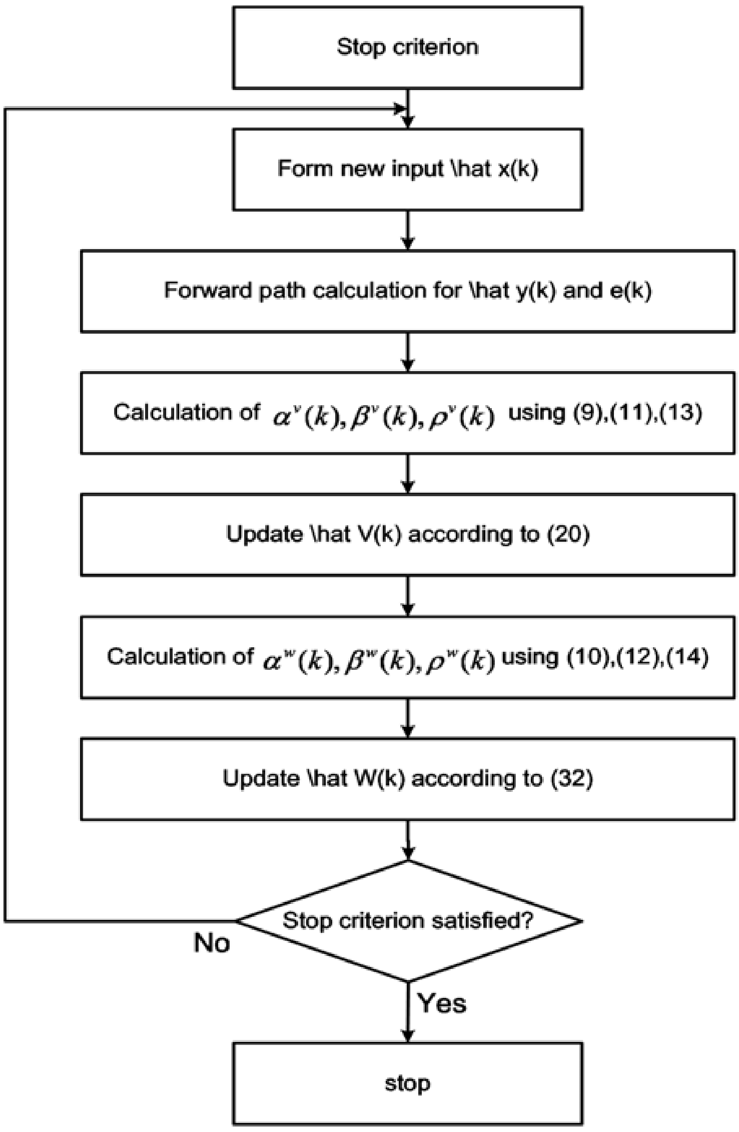

Stop criterion
Form new input \hat x(k)
Forward path calculation for \hat y(k) and e(k)
Calculation of $\alpha^v(k), \beta^v(k), \rho^v(k)$ using (9),(11),(13)
Update \hat V(k) according to (20)
Calculation of $\alpha^w(k), \beta^w(k), \rho^w(k)$ using (10),(12),(14)
Update \hat W(k) according to (32)
Stop criterion satisfied?
No
Yes
stop

Figure 4. Flow chart of the RAGD training algorithm for SISO RNN

4.1 Time series prediction

In the first simulation, the performance of the RAGD is evaluated via time series prediction problems. The RNN is employed to predict the next sample (one step) of a real sequence $\{y(k)\}$, which is generated by the following process

$$y(k+1) = \frac{y(k)y(k-1)y(k-2)u(k-1)(y(k-2)-1)+u(k)}{1+y^2(k-1)+y(k-2)} \tag{65}$$

where $u(k)$ is white Gaussian input sequence. The model of (65) is chosen from the benchmark problem in [1] (pp.159). Two data groups are generated in simulations. One is the training data set, and the other is for evaluation purpose. The traces of the time series for training and evaluation are displayed in Figure 5.

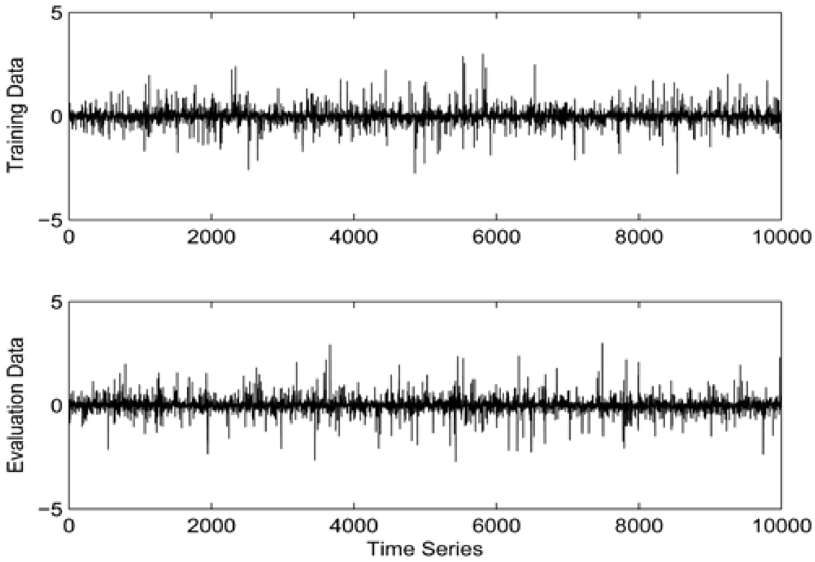


Figure 5. Sequences of the time series for training and evaluation

To provide a comparative idea, we have also implemented the N-RTRL in simulations with constant $C = 0$ and $C = 0.2$ respectively. All the simulations run for 10000 steps. In order to present a clear illustration on both transient and steady state performance of each training algorithm, the training errors are displayed by the first 100 steps and the full 10000 steps separately as shown in Figure 6 and 7. Moreover, the squared training errors of the first 100

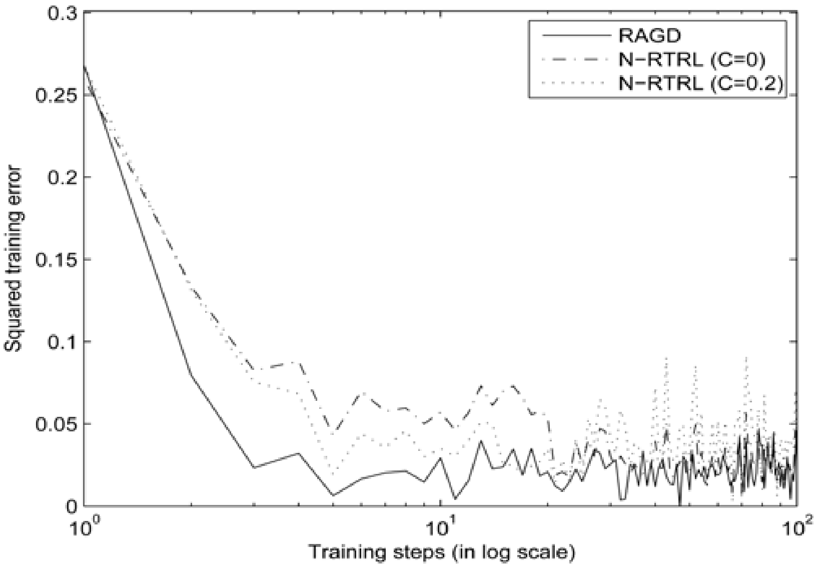


Figure 6. Squared training errors of the first 100 steps with the same set of random initializations for different algorithms

steps are plot in logarithmic format to provide a further better comparison. The steady state training errors are expressed in dB (20 times the logarithm of the amplitude ratio between error and signal) such that performance difference between the RAGD and the N-RTRL can be more explicit. The traces of the normalization factors are shown in Figure 8. The trajectories of the Frobenius norms of RNN weights with the RAGD training are displayed in Figure 9.

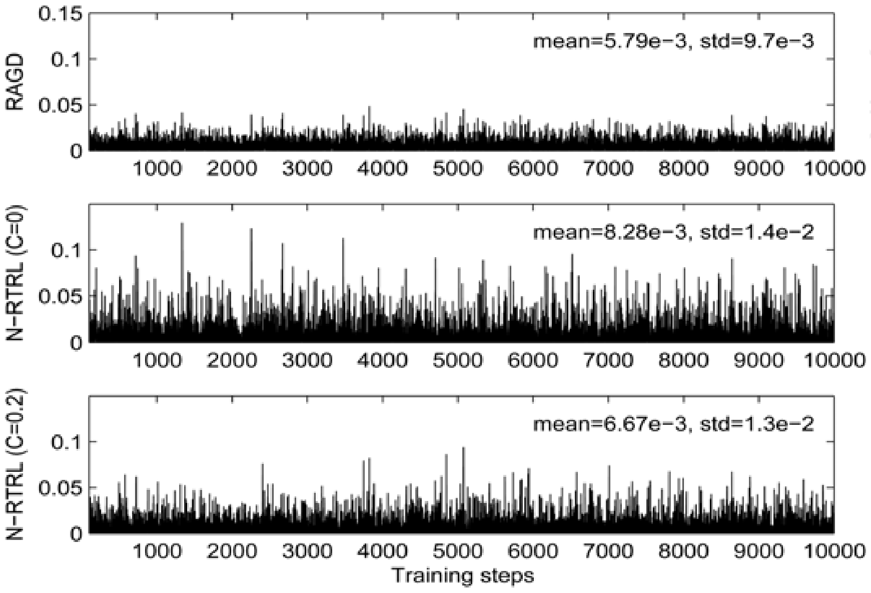


Figure 7. Squared training errors of full 3000 steps for different algorithms

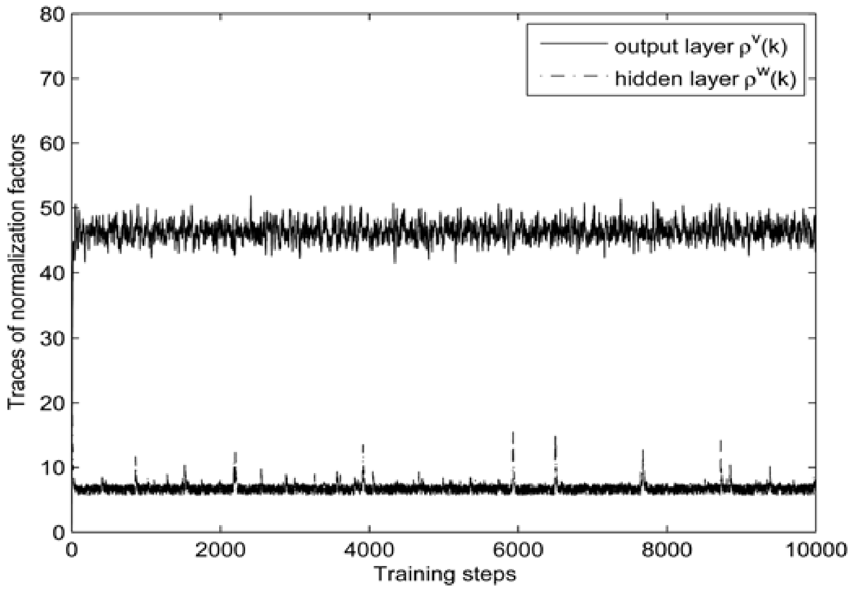


Figure 8. Traces of normalization factors $\rho^v(k)$ and $\rho^w(k)$

The results show that the RAGD algorithm is successfully stabilized in the sense that the Frobenius norms of the weights converge. The convergence of the RAGD is faster than the N- RTRL with both parameter values. Moreover, the RAGD can achieve better steady state error (mean squared training error 5.79*e*-3) than the N-RTRL (mean squared training errors 6.67*e*-3 and 8.28*e* - 3 respectively).

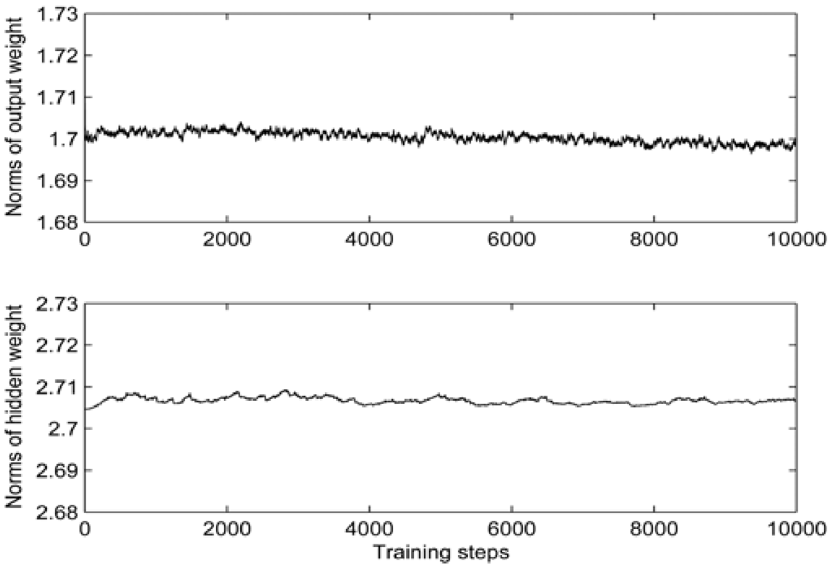


Figure 9. Traces of the Frobenius norms of RNN weights with the RAGD training

In addition to the proposed adaptive training parameters, we also investigate how the training is affected by the number of hidden layer neurons and the exponential factor of activation functions. The statistics with respect to various values of this two parameters are given in Table 1 and 2 respectively. The data are obtained by averaging the results of 50 runs (each have 10000 steps).

All simulations start with same initial weights, which can make a same starting point of the training error such that we can make a convincing comparison. The results indicate that the steady state performance is slightly improved as the λ increases. A possible reason is that transition slope of linear region of activation function becomes higher (faster) with larger λ. A similar phenomena is also observed in [7] (pp.617). In contrast, there is no obvious influence of the neuron number on the training performance.

exponential factor λ	RAGD		N-RTRL (C=0.2)		N-RTRL($C = 0$)	
	Mean	Std	Mean	Std	Mean	Std
1	5.72e-3	1.0e-2	6.61e-3	1.2e-2	8.24e-3	1.4e-2
4	5.79e-3	1.0e-2	6.67e-3	1.3e-2	8.28e-3	1.4e-2
8	5.82e-3	1.0e-2	6.73e-3	1.2e-2	8.29e-3	1.5e-2
12	5.82e-3	1.1e-2	6.75e-3	1.3e-2	8.31e-3	1.2e-2

Table 1. Statistics of squared training errors of the RAGD with different λ

number of nodes	RAGD		N-RTRL (C=0.2)		N-RTRL($C = 0$)	
	Mean	Std	Mean	Std	Mean	Std
10	5.78e-3	1.0e-2	6.69e-3	1.3e-2	8.30e-3	1.3e-2
20	5.77e-3	1.0e-2	6.64e-3	1.3e-2	8.28e-3	1.4e-2
50	5.79e-3	1.0e-2	6.67e-3	1.3e-2	8.28e-3	1.4e-2
80	5.81e-3	1.0e-2	6.68e-3	1.3e-2	8.29e-3	1.3e-2

Table 2. Statistics of squared training errors of the RAGD with di®erent neurons

4.2 Output tracking of Hammerstein-Wiener model

In the second example, the RAGD is evaluated using a system identification problem. In this simulation, the "unknown" plant consists of a dynamic linear block followed by a static nonlinearity, which is a so-called Hammerstein-Wiener model. Furthermore, the model dynamics is supposed to vary with time in terms of the time-varying coefficients of linear part, which can be expressed in a polynomial form as [19]

$$\begin{cases} x(k) = 0.3tanh(0.5u(k)) + 0.5tanh(0.8u(k)) \\ d(k) = 0.7680x(k) + 0.2872e^{-0.006k}x(k-1) - 0.1147e^{-0.03k}x(k-2) + 0.2140x(k-3) \\ \qquad -0.6435sin(0.008k)u(k-4) + 0.3548x(k-5) + 0.0197x(k-6) \\ \qquad -0.0201x(k-7) + 0.0954x(k-8) \end{cases} \tag{66}$$

The objective of the simulation is to model the plant's input-output behavior by the RNN. The command signal was given by $u(k)$, and the RNN attempts to emulate the plant output $d(k)$ as close as possible. The estimation error between actual plant output and reference signal $e(k) = d(k)-y(k)$ is fed back to RNN to adjust the weight parameters. One of the most crucial tasks in system identification is the design of appropriate excitation signals. It is important that the training data cover the entire range of plant operation due to non accurate extrapolation of RNN. In this simulation, Amplitude Modulated Pseudo Random Permutation (AMPRP) sequence are generated as training set, with the data uniformly distributed in the range of (0, 1), see Figure 10. We have also implemented the T-BPTT algorithm in simulations. The learning rate α = 0.05 (tuned by trial-and-error) was used for T-BPTT.We present the squared training error of first 1000 (transient) and 1000-5000 (steady state) steps separately in Figure 11 and 12. Results show that the RAGD converges within 200 steps while T-BPTT takes around 1000 steps. In addition, the steady state error of the RAGD is smaller than T-BPTT. Hence we say the RAGD is capable of providing a faster response to the changes of system dynamics. The traces of the normalization factors of the RAGD are provided in Figure 13.

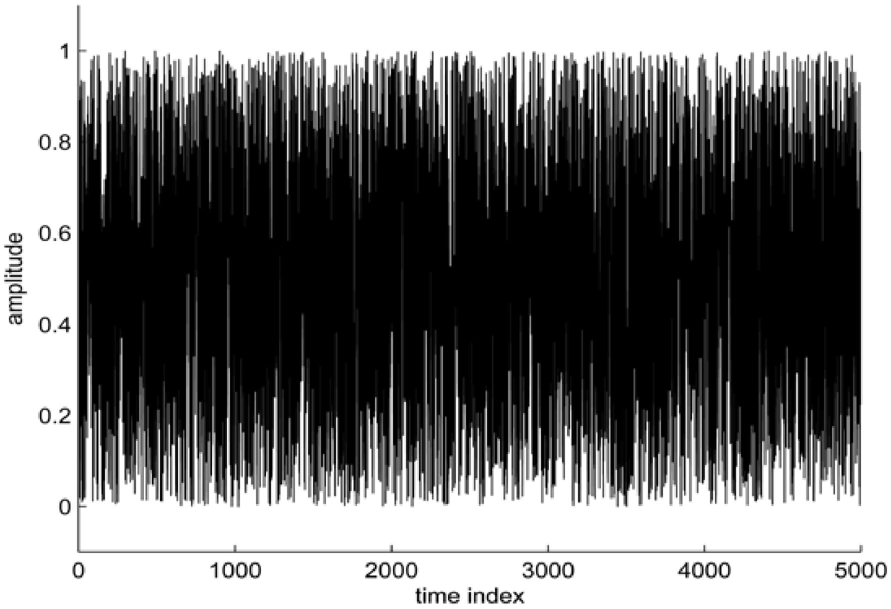


Figure 10. Trace of AMPRP input for model identification

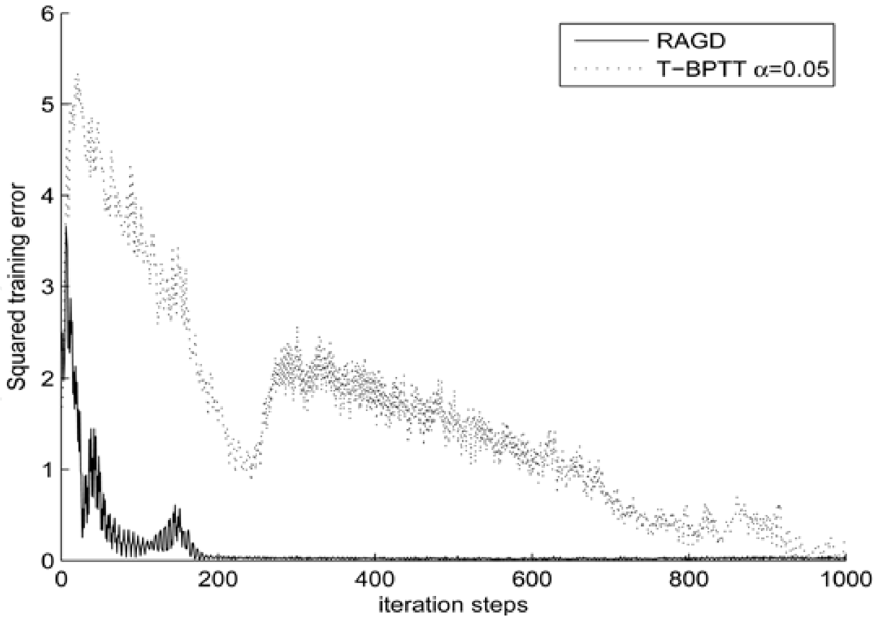


Figure 11. Squared training errors of the first 1000 steps

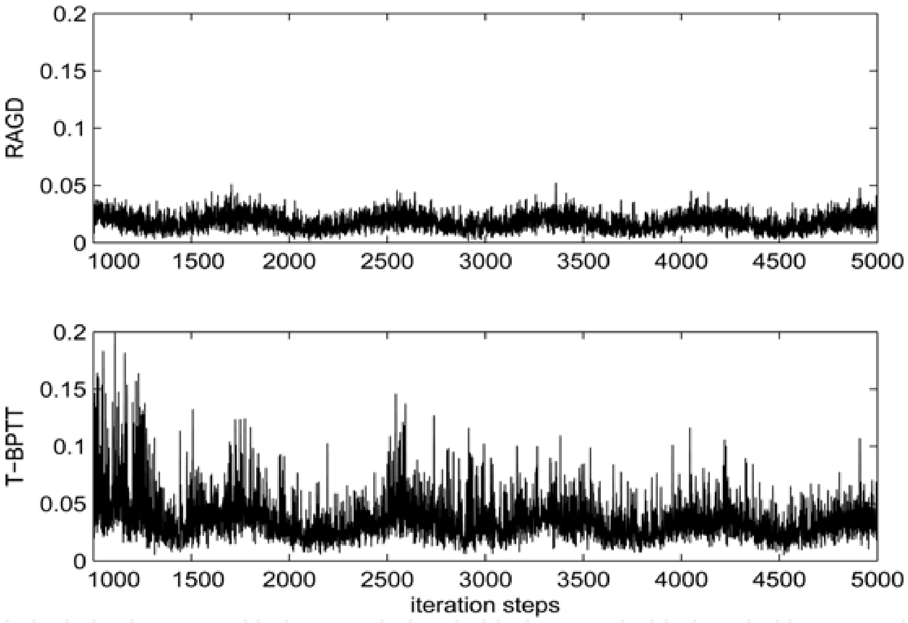


Figure 12. Squared training errors of steady state: 1000-5000 steps

4.3 Pattern association of binary image

In the last simulation, we study the problem of stable equilibrium point learning associated with a discrete-time RNN using the RAGD algorithm. In the applications of visual processing and pattern recognition, RNN plays an important role due to the feature of associative memory. The work presented in this section is inspired by an earlier paper of Liang and Gupta [8]. In [8], the authors considered absolute stability of BPTT for a general class of discrete time RNN by the Lyapunov first method. In this work the RAGD will be incorporated in place of BPTT to develop a stable learning process. To present comparison with the precedent works [20], we implement a similar simulation case of binary pattern

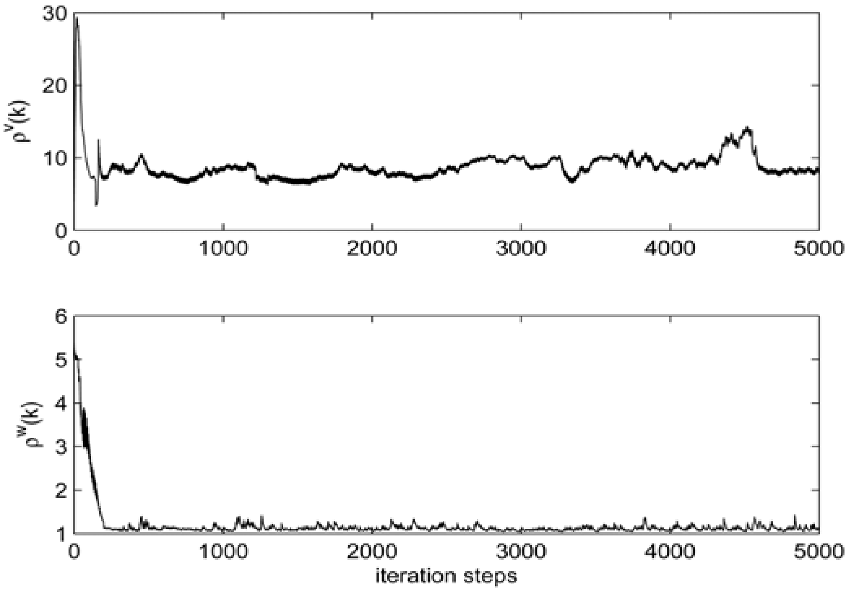


Figure 13. Traces of the normalization factors $\rho^v(k)$ and $\rho^w(k)$

association as well as BPTT algorithm, where the target pattern is a 10×10 binary image as shown in the first picture of Figure 14. The training of RNN is to store the target pattern directly as a local attractor, i.e., an equilibrium point of RNN. Since the state vector is 100 dimensional (number of pixels in target pattern) and there are no external inputs, RNN is configured with 100 inputs and outputs. As a matter of fact, this structure is analogous to the conventional Hopfield type network. RNN is utilized as an auto-associator and we aim at studying self-organizing behavior with the RAGD training algorithm. In order to demonstrate the changing of the binary image corresponding to the state of RNN during learning process, a filter layer based on sign function is added to observe the RNN output pattern, which represents the binary image at the iterative instant. The training process of the RAGD is shown in Figure 14. As mentioned, we also implement the BPTT algorithm to

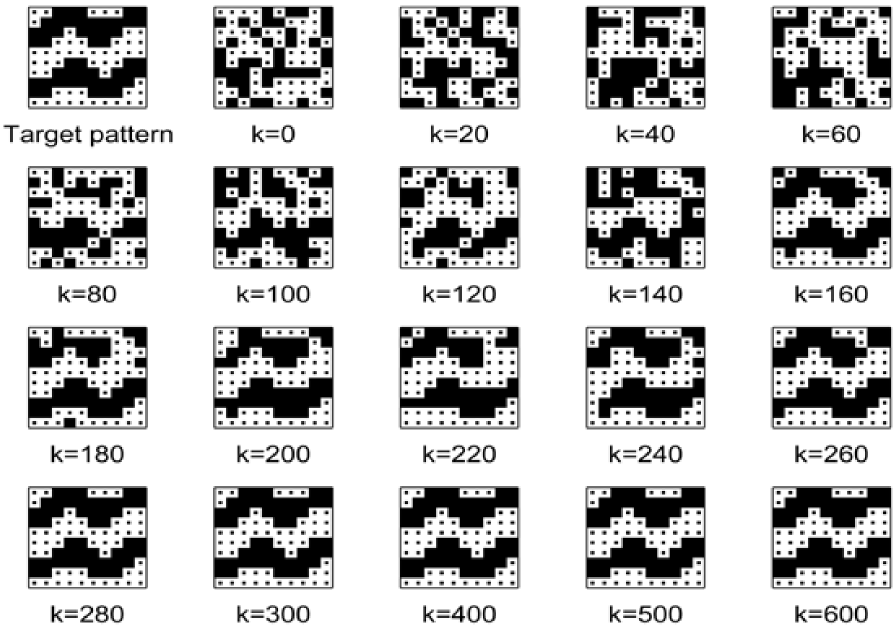


Figure 14. The binary patterns correspond to the state evolution of RNN during the training process using the RAGD algorithm.

provide comparison. The learning rate for the BPTT is 0.028. Similar to previous sections, this value is obtained by trial-and-error tuning method without violating stability constraint. The changing process of the binary image corresponding to the state vector of RNN is shown in Figure 15.

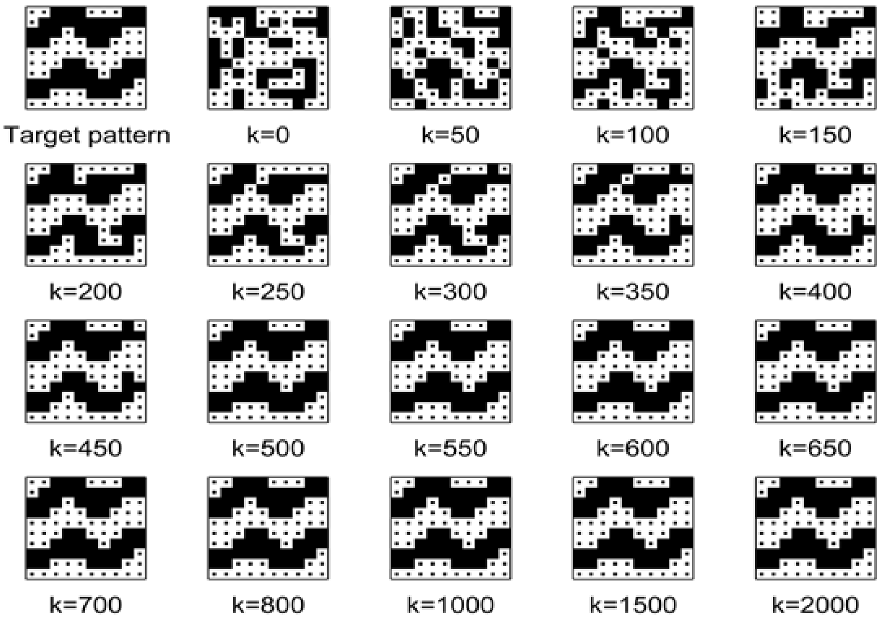


Figure 15. The binary patterns correspond to the state evolution of RNN during the training process using the BPTT algorithm.

From Figure 14 and 15, we see that using the RAGD training method, the dynamic learning process is completed within 300 steps, which is superior to the 500 steps of the BPTT algorithm. Further, we provide the squared error during the dynamic learning process of the RAGD and BPTT in Figure 16. The results indicate that the convergent process of the BPTT (about 450 iterations) is longer than the RAGD (about 280 iterations).

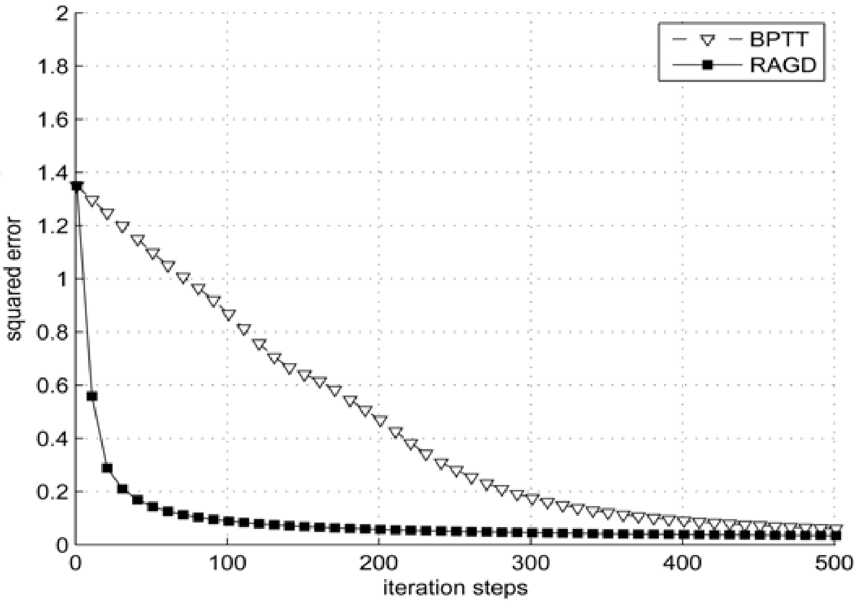


Figure 16. Comparison of the squared error curves between the RAGD and BPTT training procedures.

With these training results, we evaluate the association performance upon a distorted test pattern. The target image pattern is assumed to be disturbed by a white Gaussian noise with the noise level about 40% pixels, as shown in the first picture of Figure 17. This image is utilized as initial state of RNN to test the capability of recalling the associative memory. The recovered binary images at each time instant during recalling procedure of the two RNN trained by the RAGD and BPTT are given in Figure 17 and 18 respectively. The results show

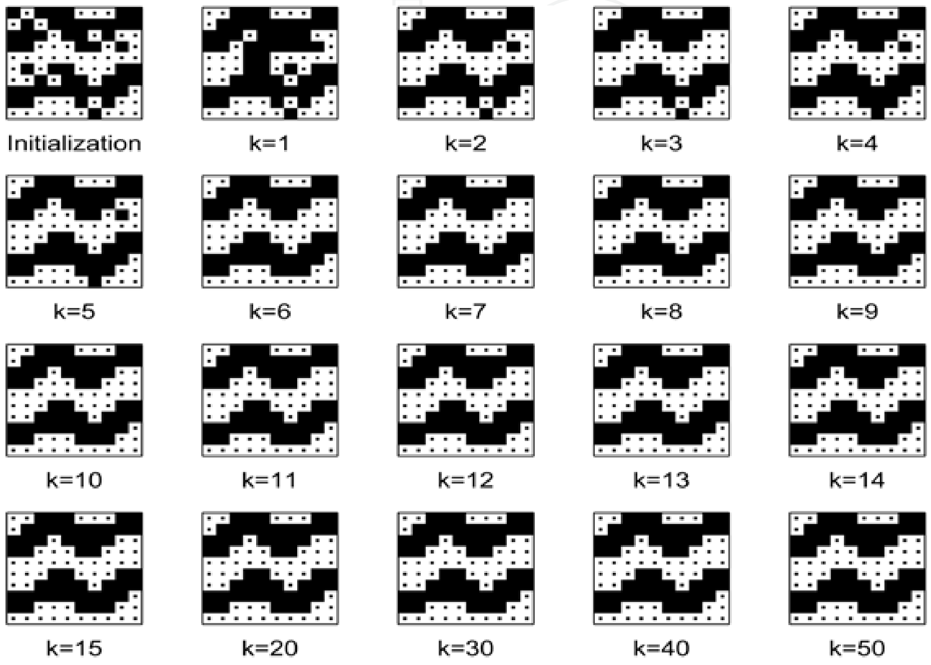


Figure 17. The binary patterns correspond to the state evolution of association process of RNN trained b the BPTT algorithm.

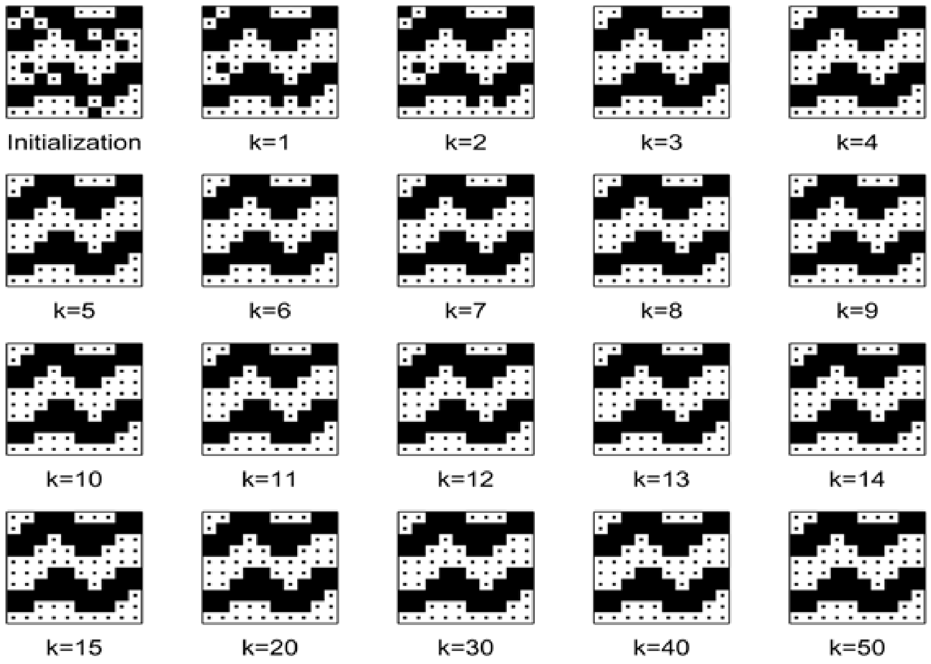


Figure 18. The binary patterns correspond to the state evolution of association process of RNN trained b the RAGD algorithm.

that the 10×10 binary pattern is successfully stored as a stable equilibrium point of the RNN by both algorithms. And there is no obvious difference of recall duration between two schemes (both within 10 iterations).

4.4 Summary

We have presented quantitative studies of the proposed RAGD algorithm in this section. Computer simulations are synthesized to justify the effectiveness of the RAGD. We give three examples which are the most frequent application areas of RNN: i) One-step prediction of non-statistical time series, which is generated by benchmark process model; ii) Identification of a nonlinear dynamic plant and the training data set is generated by a time-varying Hammerstein-Wiener model; iii) Pattern association of binary images. Further, we provide comprehensive comparisons between the RAGD and various other algorithms such as the N-RTRL, the T-BPTT, and the BPTT. In most results of these simulations, RNN trained by the RAGD demonstrates explicit advantages in the transient response speed, e.g., see Figure 6, 11 and 16. Some of the results also indicates that the RAGD can achieve better steady state responses, such as those in Figure 7 and 12. Hence by these experiment results, we conclude that the performance of the RAGD training algorithm of RNN is improved.

5. Conclusion

In this chapter, a Robust Adaptive Gradient Descent training algorithm of RNN with improved convergence speed is investigated. The major feature of the RAGD is the three adaptive parameters that switch the training patterns in a hybrid learning mode. Weight convergence and robust stability of the algorithm are analyzed via Lyapunov and input-output systematic approach respectively. We show how the training algorithm can be decomposed into a nonlinear feedforward operator H_1 and a linear feedback operator H_2, and thus form a closed loop (H_1, H_2). Then, by restricting the cone conditions of each operator, sufficient boundary conditions of L_2 stability of the training are obtained. In addition, we obtain the knowledge in which way we can adaptively change the learning rates of gradient training algorithms, or equivalently re-scale the corresponding error derivatives under stability preservation, such that the learning is ensured to be within the stable range. Such techniques are specially important for deriving a fast transient response. Another important contribution of this work lies in that we obtain a unified framework for the analysis of training algorithms of RNN by taking this systematic approach. Such an approach avoids the direct analysis of nonlinear functions in the feedforward path by applying the sector conditions. Computer simulations are also synthesized to justify the effectiveness of the RAGD. We give three examples which are most frequent application areas of RNN. The evaluation results indicate that with the proposed adaptive training parameters, the RAGD can obtain better transient and steady state responses than that of the conventional algorithms such as the BPTT, the RTRL, and the N-RTRL etc.

6. References

D. P. Mandic and J. A. Chambers, Recurrent Neural Networks for Prediction: Learning Algorithms, Architecture and Stability. *Chichester: John Wiley & Sons, 2001.*

S. Haykin, Neural Networks. *Upper saddle River, NJ: Prentice-Hall, 1999.*

M. Anthony and P. Bartlett, Neural Network Learning: Theoretical Foundations. *Cambridge, MA: Cambridge University Press, 1999.*

Y. L. Wu, Q. Song, and X. L. Yang, "Robust recurrent neural network control of biped robot," J. Intell. Robot. Syst.*, vol. 49, pp. 151-169, Jun. 2007.*

B. A. Pearlmutter, "Gradient calculations for dynamic recurrent neural networks: a survey," IEEE Trans. Neural Networks*, vol. 6, no. 5, pp. 1212-1228, 1995.*

M. Rupp and A. H. Sayed, "A time-domain feedback analysis of filtered-error adaptive gradient algorithms," IEEE Trans. Signal Process.*, vol. 44, no. 6, pp. 142-1439, 1996.*

M. Rupp and A. H. Sayed, "Supervised learning of perceptron and output feedback dynamic networks: a feedback analysis via the small gain theorem," IEEE Trans. Neural Networks, *vol. 8, no. 3, pp. 612-622, 1997.*

J. Liang and M. M. Gupta,"Stable dynamic backpropagation learning in recurrent neural networks," IEEE Trans. Neural Networks*, vol. 10, no. 6, pp. 1321-1334, 1999.*

A. F. Atiya and A. G. Parlos, "New results on recurrent network training: unifying the algorithms and accelerating convergence," IEEE Trans. Neural Networks*, vol. 11, pp. 697-709, May 2000.*

Q. Song, J. Xiao, and Y. C. Soh, "Robust backpropagation training algorithm for multi-layered neural tracking controller," IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 1133-1141, 1999.*

Q. Song, J. C. Spall, and Y. C. Soh, "Robust neural network tracking controller using simultaneous perturbation stochastic approximation," in Proc. IEEE Conf. Dec. Cont.*, vol. 42, pp. 6194-6199, Dec. 2003.*

Y. L. Wu, Q. Song, and S. Liu, "A normalized adaptive training of recurrent neural networks with augmented error gradient," IEEE Trans. Neural Networks, vol. 19, pp. 351-356, Feb. 2008.

D. P. Mandic, "Data-reusing recurrent neural adaptive filters," *Neural Comput.*, vol. 14, pp. 2693-2707, 2002.

R. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, vol. 1, pp. 270-280, 1989.

V. R. Cluett, L. Shah, and G. Fisher, \Robustness analysis of discrete-time adaptive contro systems using input-output stability theory: a tutorial," in *IEE Proc. Part-D*, vol. 135, pp. 133- 141, Mar. 1988.

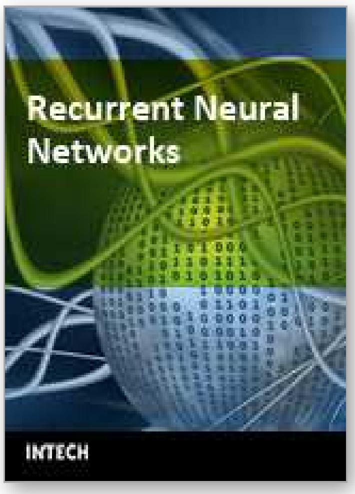
O. Nelles, *Nonlinear System Identification: From Classic Approaches to Neural Network and Fuzzy Models*. Berlin: Springer-Velag, 2001.

D. Rumelhart, G. E. Hinton, and R. Williams, "Learning internal representation by error propagation," *Parall. Distr. Process.*, vol. 1, pp. 714-728, 1986.

P. J. Werbos, "Generalisation of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, no. 1, pp. 339-356, 1988.

A. E. Nordsjo and L. H. Zetterberg, “Identification of certain time-varying nonlinear wiener and hammerstein systems," *IEEE Trans. Signal Process.*, vol. 49, pp. 577-592, Mar. 2001.

Y. L. Wu and Q. Song, “A frequency domain analysis for incremental gain of chaotic recurrent neural network," in *Proc. Int. Joint Conf. Neural Networks*, (Vancouver, BC, Canada), Jul. 2006.



Recurrent Neural Networks

Edited by Xiaolin Hu and P. Balasubramaniam

ISBN 978-953-7619-08-4

Hard cover, 400 pages

Publisher InTech

Published online 01, September, 2008

Published in print edition September, 2008

The concept of neural network originated from neuroscience, and one of its primitive aims is to help us understand the principle of the central nerve system and related behaviors through mathematical modeling. The first part of the book is a collection of three contributions dedicated to this aim. The second part of the book consists of seven chapters, all of which are about system identification and control. The third part of the book is composed of Chapter 11 and Chapter 12, where two interesting RNNs are discussed, respectively.The fourth part of the book comprises four chapters focusing on optimization problems. Doing optimization in a way like the central nerve systems of advanced animals including humans is promising from some viewpoints.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Wu Yilei, Yang Xulei and Song Qing (2008). A New Supervised Learning Algorithm of Recurrent Neural Networks and L2 Stability Analysis in Discrete-Time Domain, Recurrent Neural Networks, Xiaolin Hu and P. Balasubramaniam (Ed.), ISBN: 978-953-7619-08-4, InTech, Available from: http://www.intechopen.com/books/recurrent_neural_networks/a_new_supervised_learning_algorithm_of_recurrent_neural_networks_and_l2_stability_analysis_in_discre

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821