CONVERGENCE ANALYSIS OF THE WEIGHTED STATE SPACE SEARCH ALGORITHM FOR RECURRENT NEURAL NETWORKS

LEONG-KWAN LI
Department of Applied Mathematics
The Hong Kong Polytechnic University
Hong Kong, China

SALLY SHAO
Department of Mathematics
Cleveland State University
Cleveland, OH 44115, USA

(Communicated by Xiaoling Sun)

Abstract. Recurrent neural networks (RNNs) have emerged as a promising tool in modeling nonlinear dynamical systems. The convergence is one of the most important issues of concern among the dynamical properties for the RNNs in practical applications. The reason is that the viability of many applications of RNNs depends on their convergence properties. We study in this paper the convergence properties of the weighted state space search algorithm (WSSSA) – a derivative-free and non-random learning algorithm which searches the neighborhood of the target trajectory in the state space instead of the parameter space. Because there is no computation of partial derivatives involved, the WSSSA has a couple of salient features such as simple, fast and cost effective. In this study we provide a necessary and sufficient condition that required for the convergence of the WSSSA. Restrictions are offered that may help assure convergence of the of the WSSSA to the desired solution. The asymptotic rate of convergence is also analyzed. Our study gives insights into the problem and provides useful information for the actual design of the RNNs. A numerical example is given to support the theoretical analysis and to demonstrate that it is applicable to many applications.

1. Introduction. A recurrent neural network (RNN), which has at least one internal or external feedback loop, is modeled by the recurrently connected neurons that attempt to mimic information processing patterns in the brain. RNNs are described mathematically by dynamical systems where the states evolve according to certain nonlinear differential equations ([1]). The presence of feedback loops has a profound impact on the network learning capacity and its performance in modeling nonlinear dynamical phenomena ([12]). The dynamical properties of RNNs play an important role in their applications since their nonlinear dynamics can implement...
rich and powerful computations, allowing the RNN to perform modeling, system identification and prediction tasks for sequences with highly complex structure. Despite the great capability of RNNs, the main problems are the difficulty of training them, and the complexity of rigorously analyzing the convergence properties of the learning algorithms. The adjustments of network parameters can affect the entire RNN state variables during the network evolution ([12]) and training is usually complex and might divergent ([1]). Moreover, when the network becomes large, the training process may become prohibitively costly for each iteration even with the growing sophistication of computer hardware and software as well as mathematical algorithms.

Yiu, Wang, Teo and Tsoi in 2001 developed a novel knot-optimizing B-spline network to approximate general nonlinear system behavior ([16]). They showed that a simulated annealing algorithm with an appropriate search strategy as an optimization algorithm for the training process could avoid any possible local minima. Their work is insightful and inspiring. Li, Shao and Yiu ([7]) developed a novel learning algorithm WSSSA, a derivative-free and non-random learning algorithm, to solve the least square problems with recurrent neural dynamic constraints. The method is based on approximating a new trajectory which is a convex combination of the system output of the RNN and the desired trajectory with a ratio $\alpha$ ([7]). This approach provides the best feasible solution for the nonlinear optimization problem. Unlike the conventional gradient methods, there is no computation of partial derivatives along the target trajectory involved in the WSSSA approach, it has lower computational complexity in computing the weighted updates than the competing techniques for most typical problems.

The goal of this paper is to provide a rigorously analysis for the convergence properties of the WSSSA, which shows that the algorithm has a couple of salient features such as simple, fast learning rates and finite time convergence. Recent evidences have shown that the convergence is one of the most important issues of concern among the dynamical properties for the RNNs in practical applications. The reason is that the viability of many applications of RNNs depends on their convergence properties. Without a proper understanding of the convergence properties of RNNs, many of the applications would not be possible ([17]). We provide the necessary and sufficient conditions which are required for the convergence of the WSSSA. The restrictions are offered that may help assure convergence of the of the WSSSA to the desired solution. The asymptotic rate of convergence is also analyzed. Our result shows that the choice of the sequence of parameters depends on the choice of neural networks, which can possibly be extended for other networks, such as the B-spline network, and large-scale problems etc. We report the results of a computational experiment performed by the WSSSA algorithm, which has good exploration and exploitation capabilities in searching the optimal weight in training RNNs, and is given to support the theoretical analysis.

The organization of this paper is as follows. In section 2, we highlight the WSSSA for the discrete-time RNN model and present some the stability properties of the discrete-time RNN system. The convergence analysis for the WSSSA is provided in section 3, which the conditions that are required for convergence of the WSSSA algorithm. An empirical example is given in section 4. Some final remarks and future research directions are given in section 5.
2. The WSSSA for the discrete-time RNN model. Consider the discrete-time leaky integrator model of the RNN with $n$-neuron described by a nonlinear system of the form

$$x(t + 1) = (I - HA)x(t) + HB\sigma(Wx(t) + \theta),$$  \hspace{1cm} (1)

where $x = [x_1, x_2, ..., x_n]^T$ is a column vector with $x_i$ represents the internal state of the $i^{th}$ neuron, $\sigma$ is a neuronal activation function that is bounded, differentiable and monotonically increasing on $[-1, 1]$. $H = \text{diag}[h_1, h_2, ..., h_n]$, $A = \text{diag}[a_1, a_2, ..., a_n]$ and $B = \text{diag}[b_1, b_2, ..., b_n]$ are diagonal matrices with positive entries $h_i, a_i$ and $b_i$ for each $i = 1, 2, ..., n$, where $a_i$ represents the inverse of the neuron’s leakage time constant of $i^{th}$ neuron $\tau_i$, $b_i$ represents the neuron’s resistance, and $h_i$ is the step size of Euler’s discretization. We assume further that $\sigma(z) = \tanh(z)$ which is the symmetric sigmoid logistic function; $\theta = [\theta_1, \theta_2, ..., \theta_n]^T$ is the input bias or threshold vector of the system; $W = [w_{ij}]_{n \times n}$ is the synaptic connection weight matrix with $w_{ij}$ being the synaptic weight of a connection from neuron $n_j$ to neuron $n_i$. It is a fully connected network recurrently.

It is known that the discretized variant (1) of the continuous-time leaky integrator model of the RNN with $n$ neurons,

$$\frac{dx}{dt} = -Ax + B\sigma(Wx + \theta),$$ \hspace{1cm} (2)

will inherit the same dynamics of system (2) when the step size is “small” (see [11]). In other words, The two systems (1) and (2) share the same dynamical behavior when $h_i$ tends to 0. The positive entries of the diagonal matrix $H$ of (1) can also be used to represent the different economic cycles. It implies that if we choose appropriate $h_i’s$, the discrete-time RNN model (1) and (2) should be at least as good as any time series models ([6]). Without loss generality, we assume that for $0 < a_i h_i < 1$, $i = 1, 2, ..., n$. Since the discrete-time RNN model is easily implemented in digital hardware and easily simulated in computers, it presents advantages over the continuous-time model in practice. We focus, in this section, on the discrete-time model (1) to introduce the robust learning algorithm – the WSSSA, its stability properties and the convergence analysis of WSSSA.

Before introducing the state space search algorithm, we first discuss some stability properties of the discrete-time RNN model (1) and provide the convergence analysis afterward. The stability and bifurcation properties of the discrete-time RNNs were analyzed by Wang and Blum [11], Li [6], Jin, Nikiforuk and Gupta [3]. We introduce here some useful definitions and theorems. For the following definitions. All the norms used in this paper are either $L_2$ or $l_2$. All the results can be found in ([6]).

**Definition 2.1.** (i) A point $x^* \in [a, b]^n \subset R^n$ is defined as an equilibrium point of the discrete-time RNN system (1) if

$$x^* = (I - HA)x^* + HB\sigma(Wx^* + \theta).$$ \hspace{1cm} (3)

(ii) The function $f(x) = (I - HA)x + HB\sigma(Wx + \theta)$ has only asymptotically stable equilibrium point if all the eigenvalues of the Jacobian are inside the unit circle for all the states $x$, a given connection weight matrix $W$, and the input $\theta$.

(iii) If the discrete-time RNN system (1) has only asymptotically stable equilibrium points for a given connection weight matrix $W$ and $\theta$, then the system (1) is said to be absolutely stable.
(iv) If a sequence of vectors \( \{x_k\}_k \) converges to a limit point \( x^* \), then the asymptotic convergence rate (order of convergence) of \( \{x_k\}_k \) is defined as the supremum of the nonnegative number \( p \) satisfying

\[
0 \leq \limsup_{k \to \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} < \infty.
\]

A sequence \( \{x^*_k\}_k \) is said to be convergent linearly to \( x^* \) if

\[
0 \leq \lim_{k \to \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = \beta < 1,
\]

here \( \| \cdot \| \) denotes the \( l_2 \)-norm (the Euclidean norm) in \( \mathbb{R}^n \).

(v) A set \( A \) is called attainable of \( x \) if \( s \in A \), there exist some \( h \) such that \( x \) is reachable through \( s \) in some neighborhood of \( x \).

Remark 1. It is important to notice that the asymptotic stability may depend upon the input \( \theta \), whereas the absolute stability does not depend upon the input \( \theta \). This is the fundamental difference between the asymptotic stability and the absolute stability.

Remark 2. For the asymptotic convergence rate, larger values of the order \( p \), imply more rapid convergence.

Now we show in the following two theorems in [6] on absolute stability properties of the system (1).

**Theorem 2.2.** Let \( \beta = \max_{1 \leq i \leq n} \frac{h_i b_i}{a_i} \), then there exist at least one equilibrium point \( y^* \in [-\beta, \beta]^n \) of (2) such that

\[
y^* = A^{-1} B \sigma(W y^* + \theta)
\]

for each \( \theta \).

**Theorem 2.3.** If \( 0 < h_i a_i < 2 \) for \( i = 1, 2, \ldots, n \), and if the connection weight matrix \( W = [w_{ij}] \) satisfies the inequalities

\[
\sum_{j=1}^{n} w_{ij} < \delta_i = \frac{1}{h_i b_i} (1 - |1 - h_i a_i|)
\]

or

\[
|w_{jj}| + \frac{1}{|h_i b_i|} \sum_{i=1, i \neq j}^{n} |h_i b_i w_{ij}| < \delta_i = \frac{1}{h_i b_i} (1 - |1 - h_i a_i|)
\]

for each \( i = 1, 2, \ldots, n \), then the discrete-time RNN model (1) is absolutely stable.

Remark 3. From Theorem 2.2, we notice that inequality (5) implies that the solution space of the connection weight matrix \( W \) forms \( n \) open convex hyper cones in \( n \)-dimensional space. Moreover, \( \delta_i = \frac{1}{h_i b_i} (1 - |1 - h_i a_i|) \to \frac{\pi}{2} \) as \( h_i \to 0 \). Since the discrete-time model (1) and the continuous-time model (2) will share the same dynamical behavior as \( h_i \to 0 \), we can conclude that as \( h_i \to 0 \) both (1) and (2) are absolutely stable. It is known that for an absolutely stable neural network model, the system state will converge to one of the asymptotically stable equilibrium points regardless of the initial state.
Now given a trajectory \( y(t) \in \mathbb{R}^n \) where \( y(t) \) is a column vector and \( t = 1, \ldots, m \), we use (1) to approximate the target trajectory \( y(t) \) with the error function \( E \) defined by

\[
E(W, h, \theta) = \| x(t, W, h, \theta) - y(t) \| ^2
= \sum_{i=1}^{n} \sum_{t=1}^{m} [x_i(t, W, h, \theta) - y_i(t)]^2
\]

for some positive integer \( m \). To further simplify the notation and analysis, we fix \( h \) and let \( A = B = I_{n \times n}, i = 1, 2, \ldots, n \). We extend the trajectory dimension by one and let \( y_{n+1}(t) = 1 \) for all \( t \) in order to absorb the variable \( \theta \) into the last column of \( W \). Without loss of generality, we still consider an \( n^2 \)-dimensional constrained least square problem

\[
E(W) = \| x(t, W) - y(t) \| ^2
= \sum_{i=1}^{n} \sum_{t=1}^{m} [x_i(t, W) - y_i(t)]^2.
\] (7)

That is

\[
\min_{W} E = \min_{W} \| x(W) - y \|^2
\] (8)

subject to RNN dynamics

\[
x(t + 1) = (1 - h)x(t) + h[\sigma(Wx(t))].
\] (9)

It is well known that the gradient descent, conjugate gradient and quasi-Newton’s method have been applied to solve the least square problems for many decades. In recent years, the growing demand for sophisticated derivative free optimization methods has triggered the development of a relatively wide range of approaches (see [2]). Note that in neural networks, learning is a process of changing the network parameters \( W \) so that the system outputs \( x(t) \) will approach to the target trajectory \( y(t) \). For a fixed \( h \), in the ideal case where the network is exactly capable, that is if \( E(W_0) = 0 \), then the optimal solution can be simply solved by \( W_0 \), where

\[
W_0 = \sigma^{-1}[C_0 - D_0] - D_0 \begin{bmatrix} D_0^T (D_0 D_0^T)^{-1} \end{bmatrix},
\] (10)

\[
C_0 = [x(m), \ldots , x(2)]^T \quad \text{and} \quad D_0 = [x(m - 1), \ldots , x(1)]^T.
\] (11)

For the more general cases, naturally after we obtain \( W_0 \) from (10), we generate the whole trajectory set \( X(W_0) \) by

\[
x(t + 1) = x(t) + h[-x(t) + \sigma(W_0x(t))].
\] (12)

The basic idea of the WSSSA is that we search the class of \( x \)-homotopy in the state space \( \mathbb{R}^{m \times n} \) instead of the parameter space of \( W \) in each iteration, where \( A^+ \) is the set of attainable points of \( x \) in \( \mathbb{R}^n \). In other words, we search a reachable solution in the neighborhood of \( Y \). In the next iteration, instead of approximating the whole desired trajectory \( Y \). In many circumstances we anticipate the system
output $x(t)$ of the RNN will become closer to $y(t)$ as time goes by. That is, if $z(t) = \|x(t) - y(t)\|^2$, then $z(t)$ tends to 0 as $t$ tends to $m$. In view of this, we approximate the new set of trajectory $X_1$ defined by

$$
X_0 = X(W_0),
X_1 = \alpha_{1,t} Y + [1 - \alpha_{1,t}]X(W_0),
$$

where the weighting rule

$$
\alpha_{1,t} = \alpha_1(1 + \frac{t}{m}),
$$

where $0 < \alpha_1 < \frac{4m}{3m+1}$.

Geometrically, $\alpha_{1,t}(Y - X(W_0))$ defines the error direction. Hence, we may vary $\alpha_{1,t}$ as $\alpha_{i,t}$ in the $i$th learning iteration. By the continuity of $W$, there exists some $\alpha^*_{1,t} = \alpha_1^*(1 + \frac{t}{m})$ such that

$$
X_1 = \alpha^*_{1,t} Y + (1 - \alpha^*_{1,t})X(W_0),
$$

where $X_1$ is attainable in $C_x(A^+)$, and

$$
\{E(W_{\alpha^*_{1,t}})\} \leq E(W_0).
$$

We suppress the notation $\alpha^*_{k,t}$ as $\alpha_{k,t}$ in the rest of the paper.

In practice, we only need to store the best solution for each $\alpha_{1,t}$. Notice that $X_1$ may not be attainable even though $X(W_0)$ is attainable. However, we may repeat the process and obtain $W_1$ by

$$
W_1 = \sigma^{-1}\left[\frac{C_1 - D_1}{h} - D_1 D_1^T (D_1 D_1^T)^{-1}\right]
$$

where

$$
C_1 = [X_1(m), \cdots , X_1(2)]^T \quad \text{and} \quad D_1 = [X_1(m-1), \cdots , X_1(1)]^T.
$$

Now, we repeat the above procedure and obtain a sequence of column vectors of attainable system output $\{X(W_k)\}$ and the learning sequence $\{X_k\}$ in the state space defined by

$$
X_0 = X(W_0),
X_1 = \alpha_{1,t} Y + (1 - \alpha_{1,t})X(W_0), \quad \text{then compute $W_1$}
X_2 = \alpha_{2,t} Y + (1 - \alpha_{2,t})X(W_1), \quad \text{then compute $W_2$}
X_3 = \alpha_{3,t} Y + (1 - \alpha_{3,t})X(W_2), \quad \text{then compute $W_3$}
\vdots \quad \vdots \quad \vdots
X_{k+1} = \alpha_{k+1,t} Y + (1 - \alpha_{k+1,t})X(W_k), \quad \text{then compute $W_{k+1}$},
\quad \vdots \quad \vdots \quad \vdots
$$

where

$$
W_k = \sigma^{-1}\left[\frac{C_k - D_k}{h} - D_k D_k^T (D_k D_k^T)^{-1}\right]
$$

and

$$
C_k = [X_k(m), \cdots , X_k(2)]^T \quad \text{and} \quad D_k = [X_k(m-1), \cdots , X_k(1)]^T.
$$
To summarize the above procedures, we obtain, for each \( k \),

\[
X_0 = X(W_0), \quad X_{k+1} = \alpha_{k+1} Y + (1 - \alpha_{k+1}) X(W_k),
\]

with

\[
\alpha_{k, t} = \alpha_k \left(1 + \frac{t}{m}\right),
\]

\[
0 < \alpha_{k+1} < \frac{4m}{3m+1},
\]

\[
E(W_{k+1}) \leq E(W_k),
\]

\[
\alpha_{k+1, t} = \alpha_{k, t} \quad \text{if} \quad E(W_{k+1}) < E(W_k).
\]

The learning process stops if we obtain a solution \( W_N \) such that the error \( E(W_N) \) is less than a prescribed tolerance. Nevertheless, in practice, we store the best results for all iterations.

**Remark 4.** We may use other weighting rules, such as

\[
\alpha_{k, t} = \alpha_k f(t)
\]

where \( f(t) \) is an increasing function of \( t \) ([7]).

In the next section, we will present the convergence analysis of the WSSSA.

3. **Convergence Studies of the WSSSA.** As we mentioned in Section 1, the convergence properties of RNNs are very important for applications in optimization, image processing, and pattern recognition to name a few ([14]). The fact that the convergence of a learning algorithm is very important issue for RNNs because of their inherent signal feedback structure ([10]). For the WSSSA defined in (16), the choice of \( \alpha_{k, t} \) is an art in the center of this study. Before introduce our convergence theorem, let us introduce a useful lemma below.

**Lemma 3.1.** For each positive integer \( k \) and any nonconstant sequence \( \{\beta_k\} \) with \( 0 < \beta_k < 2 \), we have

\[
1 \leq \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j = \prod_{i=1}^{k} (1 - \beta_i).
\]

Moreover,

(i) if \( \sum_{i}^{\infty} \beta_i = \infty \), then

\[
\lim_{k \to \infty} \prod_{i=1}^{k} (1 - \beta_i) = 0,
\]

\[
\lim_{k \to \infty} \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j = 1;
\]

(ii) if \( \sum_{i}^{\infty} \beta_i < \infty \), then there exists a constant \( M \) with \( 0 < M < 1 \) such that
\begin{equation}
\lim_{k \to \infty} \prod_{i=1}^{k} (1 - \beta_i) = M, \tag{20}
\end{equation}

\begin{equation}
\lim_{k \to \infty} \left[ \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j \right] = 1 - M.
\end{equation}

\textit{Proof.} For all positive integers \(k\), let

\[ \gamma_k = \left[ \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j \right] \]

with \(\gamma_1 = \beta_1\), then

\[ \gamma_k = (1 - \beta_k)\gamma_{k-1} + \beta_k, \]

\[ (1 - \gamma_k) = (1 - \beta_k)(1 - \gamma_{k-1}), \]

where \(1 - \gamma_1 = 1 - \beta_1\). Hence

\[ 1 - \gamma_k = \prod_{i=1}^{k} (1 - \beta_i). \]

It follows that we have (18), that is,

\[ 1 - \left( \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j \right) = \prod_{i=1}^{k} (1 - \beta_i). \]

To obtain the results in (i) and (ii), we divide the proof into two cases.

(A) Case \(0 < \beta_k < 1\).

We notice that \(0 < 1 - \beta_k < e^{-\beta_k}\) for all \(0 < \beta_k < 1\), which implies

\[ 0 < \prod_{i=1}^{k} (1 - \beta_i) < \exp\left\{ - \sum_{i=1}^{k} \beta_i \right\}. \tag{21} \]

Therefore, (i) if \(\sum_{i}^{\infty} \beta_i < \infty\), then there exists a constant \(M\) with \(0 < M < 1\) such that

\[ \lim_{k \to \infty} \prod_{i=1}^{k} (1 - \beta_i) = M, \]

\[ \lim_{k \to \infty} \left[ \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j \right] = 1 - M \]

by (23), (18) and (24). Hence, (ii) if \(\sum_{i}^{\infty} \beta_i = \infty\), from (23) and (i), we have
\[ \lim_{k \to \infty} \prod_{i=1}^{k} (1 - \beta_i) = 0, \]  
\[ \lim_{k \to \infty} \left( \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j \right) = 1. \]  

Hence, Lemma 3.1 holds for the case of \( 0 < \beta_k < 1. \)

(B) Case \( 1 \leq \beta_k < 2. \)

We notice that the inequality \( 1 - \beta_k < e^{-\beta_k} \) holds for all \( 1 \leq \beta_k < 2, \) which implies

\[ \prod_{i=1}^{k} (1 - \beta_i) < \exp\{ -\sum_{i=1}^{k} \beta_i \}. \]  
(23)

For \( 1 \leq \beta_k < 2, \) \( \sum_{i}^{\infty} \beta_i = \infty \) and \( 0 \leq |1 - \beta_k| \leq 1. \) From (23) and (18), we have

\[ \lim_{k \to \infty} \prod_{i=1}^{k} (1 - \beta_i) = 0, \]  
\[ \lim_{k \to \infty} \left( \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \beta_i) \beta_j \right) = 1. \]  
(24)

The proof of Lemma 3.1 is completed. \( \square \)

We now ready prove our convergence theorem.

**Theorem 3.2.** Sequence \( \{X_k\}_k \) defined in (16) is convergent if and only if

\[ 0 < \alpha_k < \frac{4m}{3m+1} \]  
(25)

Moreover,

(a) if \( \lim_{k \to \infty} \alpha_k = \frac{2m}{3m+1}, \) then \( \lim_{k \to \infty} X_k = Y; \)

(b) if \( \sum_{k} \alpha_k \left( \frac{3m+1}{2m} \right) < \infty \) with \( 0 < \alpha_k < \frac{2m}{3m+1}, \) then \( \lim_{k \to \infty} X_k = Y + M(X_0 - Y), \)

where \( M = \lim_{k \to \infty} \prod_{i=1}^{k} (1 - \alpha_i \left( \frac{3m+1}{2m} \right)). \)

**Proof of Theorem 3.2.** Consider \( \{X_k\}_k \) defined in (16), for each \( k, \)

\[ X_{k+1} = \alpha_{k+1} \cdot Y + (1 - \alpha_{k+1} \cdot t)X(W_k) \]
\[ = \alpha_{k+1} \left( 1 + \frac{t}{m} \right) Y + [1 - \alpha_{k+1} \left( 1 + \frac{t}{m} \right)]X(W_k) \]
which implies

\[ \sum_{t} X_{k+1} = \alpha_{k+1} \frac{3m+1}{2} Y \]
\[ + [m - \alpha_{k+1} \frac{3m+1}{2}]X(W_k). \]  
(26)
By some simple algebra we obtain

\[ X_{k+1} = \alpha_{k+1}\left(\frac{3m+1}{2m}\right)Y + \left[1 - \alpha_{k+1}\left(\frac{3m+1}{2m}\right)\right]X(W_k). \] (27)

Now

\[
\|Y - X_{k+1}\| = \|Y[1 - \alpha_{k+1}\left(\frac{3m+1}{2m}\right)] - \left[1 - \alpha_{k+1}\left(\frac{3m+1}{2m}\right)\right]\|X(W_k)\|
\]

\[
= \|(1 - \alpha_{k+1}\left(\frac{3m+1}{2m}\right))\|Y - X(W_k)\|
\]

\[
\leq \|(1 - \alpha_{k+1}\left(\frac{3m+1}{2m}\right))\|Y - X_k\|
\]

\[
\leq \|(1 - \alpha_1\left(\frac{3m+1}{2m}\right))^{k+1}\|Y - X_0\|
\]

if and only if

\[ E(W_{k+1}) \leq E(W_k) \leq E(W_0), \quad 0 < \alpha_k < \frac{4m}{3m+1} \quad \text{for all } k. \]

Since sequence \(\{Y - X_k\}_k\) is monotonically decreasing and bounded below, thus, it converges, which implies that \(\{X_k\}_k\) is convergent.

Rewrite (26), we have

\[
X_k = \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \alpha_i\left(\frac{3m+1}{2m}\right))\alpha_j\left(\frac{3m+1}{2m}\right)Y
\]

\[
+ \prod_{i=1}^{k} (1 - \alpha_i\left(\frac{3m+1}{2m}\right))X(W_0). \] (29)

We notice that for \(0 < \alpha_k < \frac{4m}{3m+1}\), we have \(0 < \alpha_k\left(\frac{3m+1}{2m}\right) < 2\) for all \(k\) and positive integer \(m\). Therefore,

(a) since \(\lim_{k \to \infty} \alpha_{k+1} = \frac{2m}{3m+1}\) implies \(\sum_{k} \alpha_k\left(\frac{3m+1}{2m}\right) = \infty\). By Lemma 3.1 and (29) we obtain

\[
\lim_{k \to \infty} \prod_{i=1}^{k} (1 - \alpha_i\left(\frac{3m+1}{2m}\right)) = 0,
\]

\[
\lim_{k \to \infty} \sum_{j=1}^{k} \prod_{i=j+1}^{k} (1 - \alpha_i\left(\frac{3m+1}{2m}\right))\alpha_j\left(\frac{3m+1}{2m}\right) = 1,
\]

which implies \(\lim_{k \to \infty} X_k = Y\).

(b) If \(\sum_{k} \alpha_k\left(\frac{3m+1}{2m}\right) < \infty\) with \(0 < \alpha_k < \frac{2m}{3m+1}\), by Lemma 3.1, there exists a positive constant \(M\) with \(0 < M < 1\) such that
convergent if and only if \(0 < \alpha \leq 1\).

Our result shows that the rates of convergence of the algorithm depends on \(\alpha\). This follows from the fact that the limit point of the algorithm that provides the best feasible solution for the least square problem of Remark 6.

The WSSSA algorithm of the discrete-time RNN is a fast learning algorithm defined by (16) converges with \(\alpha = 0\) when \(\lim_{k \to \infty} X_k = Y + M(X_0 - Y)\).

We finish the proof of Theorem 3.2.

\[\lim_{k \to \infty} \prod_{i=1}^{k} \left(1 - \alpha_i \left(\frac{3m+1}{2m}\right)\right) = M, \quad \quad \lim_{k \to \infty} \sum_{j=1}^{k} \prod_{i=j+1}^{k} \left(1 - \alpha_i \left(\frac{3m+1}{2m}\right)\right) = 1 - M.\]  \hfill (30)

Lemma 3.1 and (29) implies \(\lim_{k \to \infty} X_k = Y + M(X_0 - Y)\).

Remark 5. (i) For any real sequence \(\{\alpha_k, \epsilon\}_k \subset R\), it is clear that one of the conditions

\[\sum_{k=0}^{\infty} \alpha_k \epsilon = \sum_{k} \alpha_k \left(\frac{3m+1}{2m}\right) = \infty \quad \text{and} \quad \sum_{k=0}^{\infty} \alpha_k \epsilon = \sum_{k} \alpha_k \left(\frac{3m+1}{2m}\right) < \infty \]

must hold.

(ii) The convergence analysis of Theorem 3.2 shows that \(\{X_k\}_k\) defined in (16) is convergent if and only if \(0 < \alpha_k < \frac{2m}{3m+1}\) for all \(k\). The network (9) converges to the desired solution when \(\lim_{k \to \infty} \alpha_k = \frac{2m}{3m+1}\) for \(0 < \alpha_k < \frac{4m}{3m+1}\) and the stability of the algorithm depends on \(\{\alpha_k\}\)’s. Geometrically, for each iteration, \(X_k\) defines a homotopy with respect to \(\alpha\) between the desired trajectory \(Y\) and the system output \(X(W_{k-1})\) and \(\alpha(Y - X(W_k))\) defines the error directions.

Theorem 3.3. Given \(0 < \alpha_k < \frac{4m}{3m+1}\), if \(\lim_{k \to \infty} \alpha_k = \frac{2m}{3m+1}\), the WSSSA learning algorithm defined by (16) converges with

\[\lim_{k \to \infty} \|X_k - Y\| = 0 \quad \text{and} \quad \lim_{k \to \infty} \|E(W_k)\| = 0\]

Proof of Theorem 3.3. Since the neuronal activation function \(\sigma = \sigma(z) = \tanh(z)\), \(\sigma'\) and \(\sigma''\) are continuously differentiable and uniformly bounded.

\[\lim_{k \to \infty} \alpha_k = \frac{2m}{3m+1}\] implies \(\sum_{k} \alpha_k = \infty = \sum_{k} \alpha_k \epsilon\) by Theorem 3. On the other hand, it is easy to show that

\[\lim_{k \to \infty} \sup_{k} (\alpha_k \epsilon - 1 - \alpha_{k-1} \epsilon - 1) < \infty, \quad \sum_{k} \alpha_k \epsilon^d < \infty, \quad \text{for some} \quad d > 1.\]  \hfill (31)

From our constructions, \(\{X_k\}_k\), \(\{E(W_k)\}_k\) are bounded sequences with the error sequence \(\|E(W_k)\| \downarrow 0\). Therefore, the WSSSA learning algorithm defined by (16) converges with \(\lim_{k \to \infty} \|X_k - Y\| = 0\) and \(\lim_{k \to \infty} \|E(W_k)\| = 0\), and have Theorem 3.3 holds.

Remark 6. The WSSSA algorithm of the discrete-time RNN is a fast learning algorithm that provides the best feasible solution for the least square problem of (1). This follows from the fact that the limit point of \(\{X_k\}_k\) will lead us to obtain the corresponding best feasible solution \(W\) of (1). Meanwhile, the error sequence \(\|E(W_k)\| \downarrow 0\) when \(\{X_k\}_k\) converges.

In the next theorem, we discuss the asymptotic rates of convergence of \(\{X_k\}_k\). Our result shows that the rates of convergence of \(\{X_k\}_k\) depends on \(\{\alpha_k\}\).
Theorem 3.4. If \( \lim_{k \to \infty} \alpha_k = \frac{2m}{3m+1} \) with \( 0 < \alpha_k < \frac{4m}{3m+1} \), then there exists a nonnegative number \( p_0 \in (0, 1] \) such that the asymptotic convergence rate of \( \{X_k\}_k \) is linear and equals to \( p_0 \).

Proof of Theorem 3.4. Notice that for \( 0 < \alpha_k < \frac{4m}{3m+1} \),

\[
0 \leq \frac{\|X_{k+1} - Y\|}{\|X_k - Y\|^p} = \frac{\|(1 - \alpha_{k+1})\left(\frac{3m+1}{2m}\right)X(W_k) - Y(1 - \alpha_{k+1})\left(\frac{3m+1}{2m}\right)\|}{\|X_k - Y\|^p}
\]

\[
= \frac{|(1 - \alpha_{k+1})\left(\frac{3m+1}{2m}\right)| \|X(W_k) - Y\|}{\|X_k - Y\|^p}
\]

\[
\leq \frac{|(1 - \alpha_{k+1})\left(\frac{3m+1}{2m}\right)| \|X_k - Y\|}{\|X_k - Y\|^p}
\]

\[
= |(1 - \alpha_{k+1})\left(\frac{3m+1}{2m}\right)| < 1 \quad \text{when} \quad p = 1
\]

Therefore, the order of convergence of \( \{X_k\} \) is linear. We finish the proof. \( \square \)

Remark 7. The results of Theorem 4 provides the conditions that are required for convergence of the WSSSA for the discrete-time RNN model, and Theorem 5 shows that the asymptotic rates of convergence of \( \{X_k\} \) of the WSSSA. Furthermore, \( X_k \to Y \) as \( \lim_{k \to \infty} \alpha_k = \frac{2m}{3m+1} \) with \( 0 < \alpha_k < \frac{4m}{3m+1} \).

4. The empirical example. We have tested the WSSSA learning algorithm based on the least square problem of compressing the signals of infra-red. That is to minimize the mean square error

\[
E = \sum_{i=1}^{n} \sum_{t=1}^{p} \left( x_i(t) - z((i-1)p + t) \right)^2/k^2,
\]

(33)

with the WSSSA. For a signal \( z(t) \) of length \( k \) (or with \( k \) data) is given, it is first partitioned into \( n \) equal segments with the same length \( p \); the set of neurons \( \{x_1(t), x_2(t), \ldots, x_n(t)\} \) of the RNN system \( (1) \) with network size \( n \) is fixed. We assume that the given signals are finite and continuous. We use a noise free data sequence of Cl3pheno infra-red spectrum. The size of the RNN used is \( 9 \). After the processes of segmentation, cycling and smoothing of the data, We apply the WSSSA learning method to do the approximation process to generate the compressed data, then perform the optimization process to minimize the mean square error (33). We found that the empirical results from the different samples are extremely promising. To avoid over stuff the paper we show only one of the reconstructed signals, Cl3pheno, in Figure 1 to illustrated our results. The top diagram of Figure 1 is the original signal, the bottom diagram of Figure 1 is the reconstructed signal.

The graph in Figure 2 below demonstrates the linear rate of convergence of the algorithm established in Theorem 3.4. We plot \( \frac{\|X_{k+1} - Y\|}{\|X_k - Y\|} \) against the number of iterations \( k \). We choose \( m = 1798 \), then as \( \alpha_k \to \frac{2m}{3m+1} \), we found that
Figure 1. An Infra-red signal: Cl3pheno (Top: Original signal. Bottom: Reconstructed signal).

\[ \lim_{k \to \infty} \frac{\|X_{k+1} - Y\|}{\|X_k - Y\|} = 0.010835, \]
which implies the asymptotic convergence rate of \( \{X_k\} \) is linear as we proved in Theorem 5.

Figure 2. \( \frac{\|X_{k+1} - Y\|}{\|X_k - Y\|} \to 0.10835 \) as \( k \) increasing.

In these experiments, we took the first 1800 points, choose \( m = 1798 \), use less than 500 iterations and use no averaging technique. The maximum relative error for the testing infra-red sample Cl3pheno is \( 3.3 \times 10^{-3} \). The total number of the parameters stored during the entire compression process is \( n^2 + n + 1 \) plus the 4 neglected points, where the network size \( n \). Hence, in our case, The total number of the parameters that we stored \( 9^2 + 9 + 1 = 91 \) plus the 4 neglected points.

From these experiments, we demonstrates some very promising results, which support our theoretical results in Section 3, by using the WSSSA for the RNN system to perform the data compression. They converge very fast and effective - less than 3 seconds with a MATLAB running in a reasonable current PC system.
5. **Concluding remarks and further researches.** In this paper, we investigate the derivative-free and non-random learning algorithm WSSSA, which is a learning algorithm for the nonlinear dynamical system modeling via a recurrent neural network. We provide the conditions that are required for convergence of the WSSSA algorithm in literature. The convergence analysis shows that it guarantees the convergence to the desired solution (see Theorems 3.3 and 3.4). The rates of asymptotic convergence are analyzed and the stability properties are discussed. Our studies give insights into the problem, and it can potentially beneficial to derive some new algorithms for training RNNs. The complexity analysis presented here shows that our algorithm is simpler and effective. We employed the proposed method to compress the signals and demonstrated its usefulness in performing the least square optimization for the RNN model. For other extensions to the proposed method, it will certainly be of interest to study the asynchronized recurrent networks and other neural network structures.

**Acknowledgments.** The authors would like to thank the reviewers and the editor for their valuable comments and suggestions which are very helpful for the improvement of the manuscript.

**REFERENCES**

[1] A. F. Atiya and A. G. Parlos, New results on recurrent network training: Unifying the algorithms and accelerating convergence, *IEEE Transactions on Neural Networks*, 11 (2000), 697–709.
[2] R. A. Conn, K. Scheinberg and N. L. Vicente, *Introduction to Derivative-free Optimization*, SIAM, Philadelphia, 2009.
[3] L. Jin, N. Nikifork and M. M. Gupta, Absolute stability conditions for discrete-time neural networks, *IEEE Trans. Neural Networks*, 5 (1994), 954–964.
[4] L. K. Li, Learning sunspot series dynamics by recurrent neural networks, *Advances in Data Mining and Modeling* (eds. W. K. Ching and K. P. Ng), World Science (2003), 107–115.
[5] L. K. Li and S. Shao, Dynamic properties of recurrent neural networks and its approximations, *International Journal of Pure and Applied Mathematics*, 39 (2007), 545–562.
[6] L. K. Li, S. Shao and T. Zhel, A state space search algorithm and its application to learn the short-term foreign exchange rates, *Applied Mathematical Sciences*, 2 (2008), 1705–1728.
[7] L. K. Li, Sally Shao and K. F. Cedric Yiu, Nonlinear dynamical system modeling via recurrent neural networks and a weighted state space search algorithm, *Journal of Industrial and Management Optimization*, 7 (2011), 385–400.
[8] Q. Liu and J. Wang, Finite-time convergent recurrent neural network with a hard-liming activation function for constrained optimization with piecewise-linear objective functions, *IEEE Transactions on Neural Networks*, 22 (2011), 601–613.
[9] D. T. Mirikitani and N. Nikolaev, Recursive Bayesian recurrent neural networks for time-series modeling, *IEEE Transactions on Neural Networks*, 2 (2010), 262–274.
[10] Q. Song, On the weight convergence of Elman networks, *IEEE Transactions on Neural Networks*, 21 (2010), 463–480.
[11] X. Wang and E. K. Blum, Discrete-time versus continuous-time models of neural networks, *Journal of Computer and System Sciences*, 45 (1992), 1–19.
[12] X. Wang and H. Huang, Convergence Study in Extended Kalman Filter-based Training of Recurrent Neural Networks, *IEEE Trans. on Neural Networks*, 22 (2011), 588–600.
[13] L. Xu and W. Liu, A new recurrent neural network adaptive approach for host-gate way rate control protocol within intranets using ATM ABR service, *Journal of Industrial and Management Optimization*, 1 (2005), 389–404.
[14] F. Xu and Z. Yi, Convergence Analysis of a class of simplified background neural networks with subnetworks, *Neurocomputing*, 74 (2011), 3877–3883.
[15] J. Yao and C. L. Tan, A case study on using neural networks to perform technical forecasting of forex, *Neural Computation*, 34 (2000), 79–98.
[16] K. F. C. Yiu, S. Wang, K. L. Teo and A. C. Tsoi, Nonlinear System modeling via knot-optimizing B-splines networks, *IEEE Transactions on Neural Networks*, 12 (2001), 1013–1022.
[17] Y. Zhang and K. K. Tan, *Convergence Analysis of Recurrent Neural Networks*. Kluwer, Norwell, MA, (2004).
[18] L. Zhang and Z. Yi. Selectable and unselectable sets of neurons in recurrent neural networks with saturated piecewise linear transfer function, *IEEE Transactions on Neural Networks*, 22 (2011), 1021–1031.

Received April 2013; 1st revision April 2014; final revision July 2014.

E-mail address: mailblkl@polyu.edu.hk
E-mail address: s.shao@csuohio.edu