Convergence Analysis of Three Classes of Split-Complex Gradient Algorithms for Complex-Valued Recurrent Neural Networks

Dongpo Xu
dongpoxu@gmail.com
College of Science, Harbin Engineering University, Harbin 150001,
People’s Republic of China

Huisheng Zhang
zhhuisheng@163.com
Department of Mathematics, Dalian Maritime University, Dalian 116026,
People’s Republic of China

Lijun Liu
cmljiulj@yahoo.com.cn
School of Science, Dalian Nationalities University, Dalian 116600,
People’s Republic of China

This letter presents a unified convergence analysis of the split-complex nonlinear gradient descent (SCNGD) learning algorithms for complex-valued recurrent neural networks, covering three classes of SCNGD algorithms: standard SCNGD, normalized SCNGD, and adaptive normalized SCNGD. We prove that if the activation functions are of split-complex type and some conditions are satisfied, the error function is monotonically decreasing during the training iteration process, and the gradients of the error function with respect to the real and imaginary parts of the weights converge to zero. A strong convergence result is also obtained under the assumption that the error function has only a finite number of stationary points. The simulation results are given to support the theoretical analysis.

1 Introduction

Complex-valued neural networks (CVNNs) have expanded the scope of applications to telecommunications, speech recognition, and image processing (Hirose, 2006). Compared with the real-valued neural networks, complex-valued neural networks show stronger computational power in solving classification problems (Amin & Murase, 2009). Complex-valued neural networks consist of complex valued inputs, outputs, weights, and activation functions that allow us to directly process complex valued...
signals (see, e.g., Nitta, 2009, for seismic, sonar, and radar signals). There are two commonly used complex activation functions: split-complex activation functions and fully complex activation functions. Split-complex activation functions employ a pair of real-valued activation functions to process the real and imaginary parts of a weighted sum of input signals separately, which avoids the occurrence of singular points in the adaptive learning process (Yang, Siu, & Ho, 2008). The learning algorithms for training real-valued feedforward networks, such as least mean square (LMS) algorithms and backpropagation (BP) algorithms, have been extended to complex least mean square (CLMS) algorithms (Widrow, McCool, & Ball, 1975) and complex backpropagations (CBP) (Leung & Haykin, 1991) used to train complex-valued feedforward networks. As for the complex-valued recurrent networks, Kechriotis and Manolakos (1994) and Coelho (2001) derived the complex real-time recurrent learning (CRTRL) algorithm, which uses nonlinear split-complex activation functions. A general fully complex CRTRL has been given in Goh and Mandic (2004). CRTRL is a straightforward extension of the real-valued RTRL (Williams & Zipser, 1989), which uses gradient techniques for updating the weights. Some practical algorithms, such as the adaptive step size, adaptive amplitude, data reusing, and augmented statistics, were introduced recently by Mandic and Goh (2009).

The convergence analysis of the learning algorithms is an important research topic for neural networks. Some recent deterministic convergence, weak convergence, and strong convergence results have been obtained for real-valued neural networks (Wu, Feng, Li, & Xu, 2005; Wu, Xu, & Li, 2008; Wu, Zhang, Li, & Liu, 2008; Xiong, Wu, Kang, & Zhang, 2007; Xu, Li, & Wu, 2010; Zhang, Wu, Chen, & Xiong, 2008; Zhang, Wu, Liu, & Yao, 2009). Based on the contraction mapping principle, Mandic and Chambers (2001) addressed convergence of the mean error, mean squared error, and steady state for real-valued recurrent networks. Convergence results of the gradient adaptive step size and the adaptive regularization factor for adaptive finite impulse response (FIR) filters have been obtained by Mandic, Hanna, and Razaz (2001), Hanna and Mandic (2003), Mandic (2004), and Goh and Mandic (2007). Recently Zhang, Zhang, and Wu (2009) and Zhang, Xu, and Wang (2010) studied the convergence of split-complex BP algorithms for complex-valued feedforward networks. Recurrent networks employ rich internal nonlinear dynamics, which makes them suitable for dynamic system identification and time series prediction, unlike feedforward networks, which cannot process data with time-dependent information. However, due to their complexity, the design and analysis of the learning algorithms for recurrent networks become more difficult and necessary. In this letter, we consider a finite set of complex training samples. By using a sum-of-squared-error cost function and adopting a practical definition of the split-complex activation function, three classes of split-complex nonlinear
gradient descent (SCNGD) algorithms for complex-valued recurrent neural networks (CVRNNs) are derived. To the best of our knowledge, convergence analysis of the SCNGD algorithms for CVRNNs has not yet been established in the literature, which becomes our primary concern in this letter. The theoretical analysis gives an explicit upper bound of the learning rate, which ensures the convergence of the SCNGD algorithms.

The rest of this letter is organized as follows. Some useful definitions are given in section 2. Section 3 provides a brief introduction to the fully connected recurrent neural network with a split-complex activation function. In the next three sections, we use a matrix-vector formalism to derive three classes of SCNGD algorithms. The main results are stated in section 7. Simulations are carried out in section 8 to illustrate the theory. Conclusions are given in section 9. Finally, the rigorous proofs of the main results and lemmas are presented in the appendix.

2 Some Definitions

In this section, we give some definitions that are used throughout the letter.

Definition 1. Let \( A = (a_{ij}) \) be an \( m \times n \) matrix and \( B = (b_{ij}) \) a \( p \times q \) matrix. Then the Kronecker product \( A \otimes B \) is an \( mp \times nq \) matrix defined by

\[
A \otimes B = \begin{pmatrix}
    a_{11}B & a_{12}B & \cdots & a_{1n}B \\
    a_{21}B & a_{22}B & \cdots & a_{2n}B \\
    \vdots & \vdots & \ddots & \vdots \\
    a_{m1}B & a_{m2}B & \cdots & a_{mn}B
\end{pmatrix}.
\]

Definition 2. Given an \( m \times n \) matrix \( A = (a_{ij}) \), \( \text{vec} \ A \) is defined as an \( mn \)-dimensional vector obtained by stacking the columns of the matrix \( A \) on top of one another:

\[
\text{vec} \ A = (a_{11}, a_{21}, \ldots, a_{m1}, a_{12}, a_{22}, \ldots, a_{m2}, \ldots, a_{1n}, a_{2n}, \ldots, a_{mn})^T.
\]

Definition 3. For any \( a = (a_i) \in \mathbb{R}^m \), \( g : \mathbb{R} \to \mathbb{R} \), the vector function \( G(a) \) is defined as

\[
G(a) = (g(a_1), g(a_2), \ldots, g(a_m))^T.
\] (2.1)

Further, we define \( G'(a) \) (resp. \( G''(a) \)) as a diagonal matrix with \( g'(a_1), \ldots, g'(a_m) \) (resp. \( g''(a_1), \ldots, g''(a_m) \)) as the diagonal elements.
Figure 1: Fully connected recurrent neural network.

**Definition 4.** For \( A \in \mathbb{C}^{m \times n} \) and \( z = x + iy \in \mathbb{C}^n \), where \( i = \sqrt{-1} \), \( x, y \in \mathbb{R}^n \), we define the operators as follows:

\[
\begin{align*}
re[z] &= x, \quad im[z] = y, \\
ri[z] &= \begin{pmatrix} re[z] \\ im[z] \end{pmatrix}, \quad bv[z] = \begin{pmatrix} re[z] & im[z] \\ -im[z] & re[z] \end{pmatrix}, \\
\end{align*}
\] (2.2)

The ri[·] operator has two useful properties:

\[
\begin{align*}
\|ri[z]\| &= \|z\|, \\
ri[Az] &= bm[A]ri[z] = (bv[z])^T \otimes I_m ri[\text{vec}A],
\end{align*}
\] (2.3) (2.4)

where \( \| \cdot \| \) stands for the Euclidean norm, the superscript \( T \) represents a transpose operation, and \( I_m \) denotes the \( m \times m \) identity matrix.

### 3 CVRNN with Split-Complex Activation Function

Figure 1 shows a fully connected recurrent neural network (FCRNN). It is the canonical form of a feedback neural network. The network consists of \( M \) neurons with \( N \) external inputs. We use the notation \( z^{-1} \) for unit delay. Let \( x(t) \in \mathbb{C}^N \) denote the external complex-valued input at time \( t \), \( y(t) \in \mathbb{C}^M \) the complex-valued output of processing layer, and \( y(0) = 0 \). For convenience,
we concatenate $x(t)$ and $y(t - 1)$ to form an $N + M$-dimensional vector as follows:
\[
 u(t) = \begin{pmatrix} x(t) \\ y(t - 1) \end{pmatrix}.
\] (3.1)

The network has two distinct layers: a concatenated input-feedback layer and a processing layer of computation nodes. The input and internal weights are collected into weight matrices $W_I \in \mathbb{C}^{M \times N}$ and $W_L \in \mathbb{C}^{M \times M}$, respectively. For simplicity, we write
\[
 W = (W_I, W_L) \in \mathbb{C}^{M \times (N+M)}.
\] (3.2)

The net input $s(t) \in \mathbb{C}^M$ to the processing layer is computed by
\[
 s(t) = W u(t) = W_I x(t) + W_L y(t - 1), \quad t = 1, 2, \ldots
\] (3.3)

We use the split-complex activation function suggested in Kechriotis and Manolakos (1994) and Coelho (2001). Then the output of processing layer can be expressed as
\[
 y(t) = G(\text{re}[s(t)]) + iG(\text{im}[s(t)]).
\] (3.4)

Taking the first component of the $y(t)$ as the final output of the network, we have
\[
 y_1(t) = e_1 \cdot y(t),
\] (3.5)

where $\cdot$ denotes the inner product of two vectors and $e_n$ is a vector with its $n$th component being 1 and others 0.

4 Derivation of SCNGD Algorithm

Suppose that the training sample set is $\{x(t), d(t)\}_{t=1}^Q \subset \mathbb{C}^N \times \mathbb{C}^1$, where $x(t)$ and $d(t)$ are the input and the corresponding desired output at time $t$, respectively. Let
\[
 w = \text{vec} W.
\] (4.1)

Define the following sum-of-squared error function,
\[
 E(w) = \frac{1}{2} \sum_{t=1}^Q |d(t) - y_1(t)|^2
 = \frac{1}{2} \sum_{t=1}^Q \| \text{ri}[d(t)] - \text{ri}[y_1(t)] \|^2,
\] (4.2)
2660 D. Xu, H. Zhang, and L. Liu

where $| \cdot |$ denotes the modulus of the complex number. The purpose of the network training is to find $w^*$ such that

$$E(w^*) = \min E(w).$$

(4.3)

Since $E(w)$ is a nonconstant real-valued function and its complex derivative is not defined, we compute its gradients with respect to both the real and imaginary parts of the complex weights as

$$
\begin{align*}
\nabla_r E(w) &= -Q \sum_{t=1}^{Q} \left( \frac{\partial r[y(t)]}{\partial \text{re}[w]} \right)^T (r[i[d(t)] - r[i[y_1(t)]]) \otimes e_1, \\
\nabla_i E(w) &= -Q \sum_{t=1}^{Q} \left( \frac{\partial r[y(t)]}{\partial \text{im}[w]} \right)^T (r[i[d(t)] - r[i[y_1(t)]]) \otimes e_1.
\end{align*}
$$

(4.4)

Note the convention that $(\partial \xi / \partial \xi)$ for two vectors $\xi$, and $\xi$ here is the matrix whose $(m, n)$th element is $(\partial \xi_m / \partial \xi_n)$. The factors $\frac{\partial r[y(t)]}{\partial \text{re}[w]}$ and $\frac{\partial r[y(t)]}{\partial \text{im}[w]}$ are the measures of the sensitivity of the output of the processing layer to a small variation in the value of $w$. We can get the following recursive equations for the sensitivity terms,

$$
\begin{align*}
\frac{\partial r[y(t)]}{\partial \text{re}[w]} &= G'(r[s(t)]) \left( \left( \text{re}[u(t)], \text{im}[u(t)] \right)^T \right. \\
&\left. \otimes I_M + bm[W_L] \frac{\partial r[y(t-1)]}{\partial \text{re}[w]} \right), \\
\frac{\partial r[y(t)]}{\partial \text{im}[w]} &= G'(r[s(t)]) \left( \left( -\text{im}[u(t)], \text{re}[u(t)] \right)^T \right. \\
&\left. \otimes I_M + bm[W_L] \frac{\partial r[y(t-1)]}{\partial \text{im}[w]} \right),
\end{align*}
$$

(4.5)

with the initial condition

$$
\begin{align*}
\frac{\partial r[y(0)]}{\partial \text{re}[w]} &= \frac{\partial r[y(0)]}{\partial \text{im}[w]} = 0.
\end{align*}
$$

(4.6)

Starting from an arbitrary initial value $w^0$, the weights updating rule uses a gradient descent procedure and is written as

$$w^{k+1} = w^k + \Delta w^k, \quad k = 0, 1, 2, \ldots,$$

(4.7)

$$\Delta w^k = -\eta(\nabla_r E(w^k) + i \nabla_i E(w^k)),$$

(4.8)

where $\eta \in (0, 1)$ is the learning rate.
5 Normalized SCNGD Algorithm

The normalized split-complex nonlinear gradient descent (SCNGD) algorithm can be expressed as

\[ w^{k+1} = w^k - \eta(k)(\nabla_r E(w^k) + i \nabla_i E(w^k)), \]  
\[ \eta(k) = \frac{\mu}{\sum_{t=1}^{Q} \| \frac{\partial \text{ri}[y_1(t)]}{\partial \text{ri}[w]} \|^2 + \varepsilon}. \]  

The derivation of the SCNGD algorithm starts from expressing the error function \( E(w^{k+1}) \) by a Taylor series expansion,

\[ E(w^{k+1}) = E(w^k) + \sum_{t=1}^{Q} \frac{\partial E(w^k)}{\partial \text{ri}[y_1(t)]} \text{ri}[\Delta y_1(t)] + \frac{1}{2} \sum_{t=1}^{Q} \| \Delta y_1(t) \|^2, \]  

where

\[ y_1^k(t) = y_1(w^k, t), \quad \Delta y_1^k(t) = y_1^{k+1}(t) - y_1^k(t). \]  

From equations 3.5 and 4.2, the elements of equation 5.3 are

\[ \frac{\partial E(w^k)}{\partial \text{ri}[y_1(t)]} = -(\text{ri}[d(t)] - \text{ri}[y_1^k(t)])^T, \]  
\[ \text{ri}[\Delta y_1^k(t)] = \frac{\partial \text{ri}[y_1^k(t)]}{\partial \text{ri}[w]} \text{ri}[\Delta w^k] + \alpha^k(t), \]  

where \( \alpha^k(t) \) denotes the higher-order terms of the remainder of the Taylor series expansion, equation 5.6.

Substituting equations 5.5 and 5.6 into 5.3, neglecting the higher-order terms in the Taylor series expansion, equation 5.3, and noticing that

\[ \text{ri}[\Delta w^k] = \eta \sum_{t=1}^{Q} \left( \frac{\partial \text{ri}[y_1^k(t)]}{\partial \text{ri}[w]} \right)^T (\text{ri}[d(t)] - \text{ri}[y_1^k(t)]), \]  

we obtain

\[ E(w^{k+1}) \approx E(w^k) - \left( \frac{1}{\eta} - \sum_{t=1}^{Q} \| \frac{\partial \text{ri}[y_1^k(t)]}{\partial \text{ri}[w]} \|^2 \right) \| \Delta w^k \|^2. \]  

This gives the following optimal learning rate for the SCNGD algorithm,

\[ \eta_{\text{opt}}(k) = \frac{1}{\sum_{t=1}^{Q} \| \frac{\partial \text{ri}[y_1^k(t)]}{\partial \text{ri}[w]} \|^2 + \varepsilon}. \]
where \( \varepsilon \) denotes a term added to compensate the exclusion of second- and higher-order terms from the Taylor series expansion. Although \( \varepsilon \) is varying in time, this term was chosen, for simplicity, to be a constant in the SCNNGD algorithm.

6 Adaptive SCNNGD Algorithm

Following the approach of Hanna and Mandic (2003) and Mandic (2004), consider an adaptive SCNNGD algorithm:

\[
\mathbf{w}^{k+1} = \mathbf{w}^k - \eta(k)(\nabla_{\mathbf{w}} E(\mathbf{w}^k) + i \nabla_{\mathbf{w}} E(\mathbf{w}^k)),
\]

\[
\eta(k) = \frac{\mu}{\sum_{t=1}^{Q} \left\| \frac{\partial r_{\mathbf{y}}(t)}{\partial r_{\mathbf{w}}} \right\|^2 + \varepsilon(k)}.
\]

The regularization parameter \( \varepsilon(k) \) is updated according to

\[
\varepsilon(k) = \varepsilon(k - 1) - \rho(k) \nabla_{\varepsilon} E(\mathbf{w}^k)|_{\varepsilon=\varepsilon(k-1)},
\]

where \( \nabla_{\varepsilon} E(\mathbf{w}^k) \) denotes the gradient of the error function with respect to the regularization factor \( \varepsilon \) and \( \rho(k) \) denotes the variable step size, which satisfies \( \rho(k) = O(1/k) \).

Writing the weight update term dividing the learning rate,

\[
\Psi(k) = \frac{\mathbf{r}_{\mathbf{i}}[\Delta \mathbf{w}^k]}{\eta(k)} = \left( \frac{\partial E(\mathbf{w}^k)}{\partial r_{\mathbf{i}}[\mathbf{w}]} \right)^T
\]

\[
= \sum_{t=1}^{Q} \left( \frac{\partial r_{\mathbf{i}}[y_{\mathbf{i}}^k(t)]}{\partial r_{\mathbf{i}}[\mathbf{w}]} \right)^T (r_{\mathbf{i}}[d(t)] - r_{\mathbf{i}}[y_{\mathbf{i}}^k(t)]),
\]

and denoting \( \gamma(k) = \partial r_{\mathbf{i}}[\mathbf{w}^k]/\partial \varepsilon(k-1) \), we can derive

\[
\nabla_{\varepsilon} E(\mathbf{w}^k)|_{\varepsilon=\varepsilon(k-1)} = (\Psi(k))^T \gamma(k). \quad (6.5)
\]

From equation 6.1, the derivative of the weight \( \mathbf{w}^k \) with respect to \( \varepsilon(k-1) \) becomes

\[
\gamma(k) = \frac{\mu \Psi(k - 1)}{(\sum_{t=1}^{Q} \left\| \frac{\partial r_{\mathbf{i}}[y_{\mathbf{i}}^{k-1}(t)]}{\partial r_{\mathbf{i}}[\mathbf{w}]} \right\|^2 + \varepsilon(k - 1))^2}.
\]
This yields the update of regularization factor in the form
\[
\varepsilon(k) = \varepsilon(k-1) - \mu \rho(k) \frac{(\Psi(k))^T \Psi(k-1)}{\left( \sum_{t=1}^{Q} \left\| \frac{\partial r_i[y_{k-1}]}{\partial \mathbf{w}} \right\|_2^2 + \varepsilon(k-1) \right)^2},
\]  
(6.7)

The adaptive normalized nonlinear split-complex gradient descent (SCAN-NGD) algorithm can now be summarized as
\[
\mathbf{w}^{k+1} = \mathbf{w}^k - \eta(k) (\nabla_r E (\mathbf{w}^k) + i \nabla_i E (\mathbf{w}^k)),
\]  
(6.8)
\[
\eta(k) = \frac{\mu}{\sum_{t=1}^{Q} \left\| \frac{\partial r_i[y_{k}}{\partial \mathbf{w}} \right\|_2^2 + \varepsilon(k)},
\]  
(6.9)
\[
\varepsilon(k) = \varepsilon(k-1) - \mu \rho(k) \frac{(\Psi(k))^T \Psi(k-1)}{\left( \sum_{t=1}^{Q} \left\| \frac{\partial r_i[y_{k-1}]}{\partial \mathbf{w}} \right\|_2^2 + \varepsilon(k-1) \right)^2},
\]  
(6.10)

7 Convergence Analysis

The following assumptions will be used:

A1. \(|G(a)|, |G'(a)|, |G''(a)|\) are bounded for any \(a \in \mathbb{R}^m\).

A2. \(|\mathbf{W}_k| (k = 0, 1, 2, \ldots)\) (see equations 3.2 and 4.1) are bounded in the learning process, equation 4.7.

A3. There exists a closed bounded region \(\Phi \subset \mathbb{C}^{M(N+M)}\) such that \(\{\mathbf{w}^k\}\) \(\subset \Phi\), and the set \(\Phi_0 = \{\mathbf{w} \in \Phi : \nabla_r E(\mathbf{w}) = 0, \nabla_i E(\mathbf{w}) = 0\}\) contains only a finite number of points.

Remark 1. In definition 3, the real-valued function \(g\) is typically a sigmoid function; thus, condition A1 can be satisfied. In order to guarantee the weak convergence, a condition like A2 is often used in literature (Gori & Maggini, 1996); this condition can be removed when a gradient learning method includes a penalty term (Zhang, Wu, et al., 2009). Condition A3 implies that the error function has only a finite number of local minimums, which is used to obtain the strong convergence.

Now we give the main results. Detail of the proofs are in the appendix to make the presentation here more readable.

Theorem 1. Let assumptions A1 and A2 be valid. For the SCNGN algorithms, 4.7 and 4.8, denote the weight sequence starting from arbitrary \(\mathbf{w}^0\) by \(\mathbf{w}^k\). If the learning rate \(\eta\) satisfies equation A.13, we have:
a. \[ E(w^{k+1}) \leq E(w^k), \quad k = 0, 1, 2, \ldots \]

b. \[ \lim_{k \to \infty} \| \nabla_r E(w^k) \| = 0, \quad \lim_{k \to \infty} \| \nabla_i E(w^k) \| = 0. \]

Furthermore, if assumption A3 also is valid, then there exists a point \( w^* \in \Phi_0 \) such that

c. \[ \lim_{k \to \infty} w^k = w^*. \]

**Theorem 2.** The same conclusions as in theorem 1 are valid for SCNNGN and SCANNGN algorithms.

Statement a of theorem 1 shows the monotonicity of the error function \( E(w) \) in the learning iteration process. Statement b confirms the weak convergence of the weight sequence \( \{w^k\} \), that is, the gradients of the error function with respect to the real and imaginary parts of the weights converge to zero. Statement c presents a strong convergence, that is, the sequence \( \{w^k\} \) itself converges if the error function has only a finite number of stationary points.

## 8 Simulations

We illustrate the convergence behavior of the SCNNG algorithms by using two numerical examples. The nonlinearities within neurons were chosen to be the split-complex sigmoid function \( \Phi(z) = \frac{1}{1+e^{-\beta z} + i \frac{1}{1+e^{-\beta z}}} \), where \( z = x + iy \in \mathbb{C} \) and \( \beta = 1 \). The architecture of the FCRNN (see Figure 1) consisted of \( M = 10 \) neurons, with the external input length of \( N = 4 \). The real and imaginary parts of the initial weights \( w^0 \) were randomly chosen from the uniform distribution in the range \([-1, 1]\), and the training procedure was epochwise, with 2000 epochs consisting of \( Q = 20 \) samples. After each epoch, the mean squared prediction error (MSE) was calculated for all training samples.

Simulations were undertaken by averaging 10 iterations of independent trials on the one-step-ahead prediction of both the complex-valued colored and nonlinear signals. The colored signal was a stable complex \( AR(4) \) process given by

\[
    r(t) = 1.79r(t - 1) - 1.85r(t - 2)
    + 1.27r(t - 3) - 0.41r(t - 4) + n(t),
\]

where \( n(t) \) was a complex white gaussian noise (CWGN) with zero mean and unit variance. The nonlinear benchmark input signal was (Mandic & Goh, 2009)

\[
    z(t) = \frac{z(t - 1)}{1 + z^2(t - 1)} + n^3(t).
\]
Figure 2: Learning curves of SCNGD for the colored input (see equation 8.1) and nonlinear input (see equation 8.2) obtained by using $\mu = 1$ and $\eta = 0.1$, 0.01, and 0.001. (a) Colored signal. (b) Nonlinear signal.

Figure 3: Time variation of $\|\Psi(k)\|$ for SCNGN on the colored (see equation 8.1) and nonlinear (see equation 8.2) signals for $\mu = 1$ and various learning rate $\eta$. (a) Colored signal. (b) Nonlinear signal.

Figure 2 shows the averaged performance curves over 10 iterations of independent trials for the SCNGN algorithm on complex colored (see equation 8.1) and nonlinear (see equation 8.2) signals, plotted for different small learning rates, indicating that for a small learning rate, decreasing the learning rate of SCNGN will cause the monotonicity of MSE during the training iteration process, although with slower initial convergence. Figure 3 illustrates that the norm of $\Psi(k)$ (see equation 6.4) converges to zero in magnitude along the iterations. Figures 4 and 5 show the performance of the SCNNNG algorithm with the compensation term $\varepsilon = 10$, 100,
and 1000 for predictions of colored (see equation 8.1) and nonlinear (see equation 8.2) input. It is shown that the averaged MSE decreases monotonically and, correspondingly the gradient $\Psi(k)$ converges to zero when an appropriate constant $\epsilon$ is chosen. Figures 6 and 7 show the learning curves of the SCANNGN algorithm on colored input (see equation 8.1) and nonlinear input (see equation 8.2) using $\rho(k) = 1/k$ and different initial value $\epsilon(0)$, which indicates that a large $\epsilon(0)$ can guarantee convergence of the SCANNGN algorithm. Thus, the simulation results clearly support our convergence theorems in section 7.
In this letter, we have studied the convergence of split-complex nonlinear gradient learning algorithms for complex-valued recurrent neural networks with split-complex transfer functions. We have provided an upper bound of the learning rate and convergence conditions, which ensure monotonicity of the error function and the convergence of the gradient sequence of the error function. When the error function has a finite number of stationary points, the weight sequence is proved to be convergent. The results obtained here
are helpful in designing the complex-valued neural networks and choosing the appropriate learning rate.

Appendix: Proofs of Results

The convergence proof for SCNGN is presented in section A.1. In sections A.2 and A.3, we briefly point out how to extend the result to SCNNGN and SCANNGN algorithms, respectively.

A.1 Convergence Analysis for SCNGN. We show a few lemmas before presenting the proof of theorem 1. We use the following notations:

\[ u^k(t) = u(w^k, t) \] (A.1)

\[ s^k(t) = s(w^k, t), \quad \Delta s^k(t) = s^{k+1}(t) - s^k(t) \] (A.2)

\[ y^k(t) = y(w^k, t), \quad \Delta y^k(t) = y^{k+1}(t) - y^k(t) \] (A.3)

\[ \prod_{l=n}^m A_l = \begin{cases} A_mA_{m-1} \cdots A_n, & n \leq m \\ I, & n > m \end{cases} \] (A.4)

Lemma 1. For the iteration process, equation 4.5, and initial condition 4.6,

\[ \frac{\partial r^i[y^k(t)]}{\partial r^i[w]} = \sum_{n=0}^{t-1} \prod_{l=1-n}^t (G'(r^i[s^k(l)]) bm[W^k_L]) \times G'(r^i[s^k(t-n)])(bv[u^k(t-n)])^T \otimes I_M. \] (A.5)

Proof. The proof is done by induction. Using equations 2.2, 4.5, and 4.6, we have

\[ \frac{\partial r^i[y^k(1)]}{\partial r^i[w]} = G'(r^i[s^k(1)])(bv[u^k(1)])^T \otimes I_M, \]

which is equation A.5 when \( t = 1 \), and this starts the induction.

Now suppose that equation A.5 is true for some positive integer \( t \). Integrate by parts,

\[ \frac{\partial r^i[y^k(t+1)]}{\partial r^i[w]} = G'(r^i[s^k(t+1)])(bv[u^k(t+1)])^T \otimes I_M \]

\[ + G'(r^i[s^k(t+1)])bm[W^k_L] \frac{\partial r^i[y^k(t)]}{\partial r^i[w]} \]
\begin{align*}
&= G'(ri[s^k(t+1)])(bv[u^k(t+1)])^T \otimes I_M \\
&\quad + \sum_{n=0}^{t-1} \prod_{l=t-n+1}^{t+1} (G'(ri[s^k(l)]) bm[W^k_L]) \\
&\quad \times G'(ri[s^k(t-n)])(bv[u^k(t-n)])^T \otimes I_M \\
&\quad = \sum_{n=1}^{t-1} \prod_{l=t-n+1}^{t+1} (G'(ri[s^k(l)]) bm[W^k_L]) \\
&\quad \times G'(ri[s^k(t-n)])(bv[u^k(t-n)])^T \otimes I_M \\
&\quad = \sum_{n=0}^{t} \prod_{l=t-n+2}^{t+1} (G'(ri[s^k(l)]) bm[W^k_L]) G'(ri[s^k(t-n+1)]) \\
&\quad \times (bv[u^k(t-n+1)])^T \otimes I_M, \\
\end{align*}

which is equation A.5 with \( t \) replaced by \( t + 1 \), and this is the inductive step. Thus, by induction, equation A.5 holds for all positive integers \( t \).

**Lemma 2.** For the terms \( s^k(t) \) and \( y^k(t) \) given in equations A.2 and A.3, there exists a first-order Taylor series expansion,

\begin{align*}
ri[\Delta y^k(t)] &= G'(ri[\sigma^k(t)]) ri[\Delta s^k(t)], \\
ri[\Delta s^k(t)] &= \sum_{n=0}^{t-1} \prod_{l=t-n+1}^{t} (bm[W^k_{L+1}]) G'(ri[\sigma^k(l-1)]) \\
&\quad \times ((bv[u^k(t-n)])^T \otimes I_M) ri[\Delta w^k],
\end{align*}

where \( ri[\sigma^k(t)] \in \mathbb{R}^{2M} \) is a real-valued vector between \( ri[s^k(t)] \) and \( ri[s(u^{k+1}, t)] \) in an element-by-element manner.

**Proof.** The proof is done by induction. For the base case \( t = 1 \), the left-hand side is

\begin{align*}
ri[\Delta s^k(1)] &= ri[W^{k+1}u^{k+1}(1)] - ri[W^ku^k(1)] = ri[W^{k+1}u^{k}(1)] - ri[W^ku^{k}(1)] \\
&= ((bv[u^{k}(1)])^T \otimes I_M) ri[\text{vec}W^{k+1}] - ((bv[u^{k}(1)])^T \otimes I_M) ri[\text{vec}W^k] \\
&= ((bv[u^{k}(1)])^T \otimes I_M) ri[\Delta w^k].
\end{align*}

which is equation A.7 when \( t = 1 \), and this starts the induction.
Now assume that equation A.7 is true for some positive integer $t$. Integrated by parts,
\[
\begin{align*}
\text{ri}[\Delta s^k(t + 1)] & = \text{ri}[W^{k+1}u^{k+1}(t + 1)] - \text{ri}[W^ku^k(t + 1)] \\
& = \text{ri}[W^{k+1} - W^k]u^k(t + 1) \\
& \quad + bm[W^{k+1}](\text{ri}[u^{k+1}(t + 1)] - \text{ri}[u^k(t + 1)]) \\
& = ((bv[u^k(t + 1)])^T \otimes I_M) \text{ri}[\Delta w^k] \\
& \quad + bm[W^{k+1}](\text{ri}[y^{k+1}(t)] - \text{ri}[y^k(t)]) \\
& = ((bv[u^k(t + 1)])^T \otimes I_M) \text{ri}[\Delta w^k] \\
& \quad + \sum_{n=0}^{t-1} \prod_{l=n+1}^{t+1} (bm[W^{k+1}_l]) G'(\text{ri}[\sigma^k(l - 1)]) \\
& \quad \times ((bv[u^k(t - n)])^T \otimes I_M) \text{ri}[\Delta w^k] \\
& = \sum_{n=-1}^{t-1} \prod_{l=n+1}^{t+1} (bm[W^{k+1}_l]) G'(\text{ri}[\sigma^k(l - 1)]) \\
& \quad \times ((bv[u^k(t - n)])^T \otimes I_M) \text{ri}[\Delta w^k] \\
& = \sum_{n=0}^{t-1} \prod_{l=n+2}^{t+1} (bm[W^{k+1}_l]) G'(\text{ri}[\sigma^k(l - 1)]) \\
& \quad \times ((bv[u^k(t - n + 1)])^T \otimes I_M) \text{ri}[\Delta w^k].
\end{align*}
\]
which is equation A.7, with $t$ replaced by $t + 1$, and this is the inductive step. Thus by induction, equation A.7 holds for all positive integers $t$.

**Lemma 3.** For the term $y^k(t)$ given in equation A.3, we have a second-order Taylor series expansion,
\[
\text{ri}[\Delta y^k(t)] = \frac{\partial \text{ri}[y^k(t)]}{\partial \text{ri}[w]} \text{ri}[\Delta w^k] + \delta^k(t),
\]
(A.8)
where
\[
\delta^k(t) = \sum_{n=1}^{t-1} \prod_{l=n+1}^{t-1} \left( G'(\text{ri}[s^k(l + 1)]) bm[W^{k+1}_l] \right) G'(\text{ri}[s^k(t - n + 1)]) \\
\times bm[\Delta W^k] \text{ri}[\Delta y^k(t - n)]
\]
argue by induction. For the base case and the right-hand side is
\[ ri[\tau^k(t)] \text{ is true, which starts the induction.} \]

Now assume that \( ri[\tau^k(t)] \) is true for some positive integer \( t \). We relate \( P(t) \) to \( P(t+1) \):

\[
ri[\Delta y^k(t+1)] \]
\[ = G'(ri[s^k(t+1)]) (ri[\Delta s^k(t+1)]) \\
+ \frac{1}{2} G''(ri[\tau^k(t+1)]) (ri[\Delta s^k(t+1)])^{(2)} \\
= G'(ri[s^k(t+1)]) bm[W_L^k] ri[\Delta y^k(t)] \\
+ G'(ri[s^k(t+1)]) (bv[u^k(t+1)])^T \otimes I_M ri[\Delta w^k] \\
+ G'(ri[s^k(t+1)]) bm[\Delta W_L^k] ri[\Delta y^k(t)] \\
+ \frac{1}{2} G''(ri[\tau^k(t+1)]) (ri[\Delta s^k(t+1)])^{(2)} \\
= G'(ri[s^k(t+1)]) bm[W_L^k] \frac{\partial ri[y^k(t)]}{\partial ri[w]} ri[\Delta w^k] \\
+ G'(ri[s^k(t+1)]) bm[W_L^k] \delta^k(t) \\
+ G'(ri[s^k(t+1)]) (bv[u^k(t+1)])^T \otimes I_M ri[\Delta w^k] \\
+ G'(ri[s^k(t+1)]) bm[\Delta W_L^k] ri[\Delta y^k(t)] \\
+ \frac{1}{2} G''(ri[\tau^k(t+1)]) (ri[\Delta s^k(t+1)])^{(2)} \\
= \frac{\partial ri[y^k(t+1)]}{\partial ri[w]} ri[\Delta w^k] + \delta^k(t+1),
\]
and each component of \( ri[\tau^k(t)] \) lies between the two corresponding components of \( ri[s^k(t)] \) and \( ri[s^{k+1}(t)] \).

**Proof** Let \( P(t) \) be the proposition that equation A.8 holds for all \( t \in N \). We argue by induction. For the base case \( t = 1 \), the left-hand side is

\[
ri[\Delta y^k(1)] = G'(ri[\tau^k(1)]) ri[\Delta s^k(1)] \\
= G'(ri[\tau^k(1)]) (bv[u^k(1)])^T \otimes I_M ri[\Delta w^k],
\]

and the right-hand side is \( G'(ri[\tau^k(1)]) (bv[u^k(1)])^T \otimes I_M ri[\Delta w^k] \), so \( P(1) \) is true, which starts the induction.

Now assume that \( P(t) \) is true for some positive integer \( t \). We relate \( P(t+1) \) to \( P(t) \):

\[
ri[\Delta y^k(t+1)] \\
= G'(ri[s^k(t+1)]) ri[\Delta s^k(t+1)] \\
+ \frac{1}{2} G''(ri[\tau^k(t+1)]) ri[\Delta s^k(t+1)]^{(2)} \\
= G'(ri[s^k(t+1)]) bm[W_L^k] ri[\Delta y^k(t)] \\
+ G'(ri[s^k(t+1)]) (bv[u^k(t+1)])^T \otimes I_M ri[\Delta w^k] \\
+ G'(ri[s^k(t+1)]) bm[\Delta W_L^k] ri[\Delta y^k(t)] \\
+ \frac{1}{2} G''(ri[\tau^k(t+1)]) ri[\Delta s^k(t+1)]^{(2)} \\
= G'(ri[s^k(t+1)]) bm[W_L^k] \frac{\partial ri[y^k(t)]}{\partial ri[w]} ri[\Delta w^k] \\
+ G'(ri[s^k(t+1)]) bm[W_L^k] \delta^k(t) \\
+ G'(ri[s^k(t+1)]) (bv[u^k(t+1)])^T \otimes I_M ri[\Delta w^k] \\
+ G'(ri[s^k(t+1)]) bm[\Delta W_L^k] ri[\Delta y^k(t)] \\
+ \frac{1}{2} G''(ri[\tau^k(t+1)]) ri[\Delta s^k(t+1)]^{(2)} \\
= \frac{\partial ri[y^k(t+1)]}{\partial ri[w]} ri[\Delta w^k] + \delta^k(t+1),
\]
which is \( P(t+1) \), completing the inductive step. Thus, by the principle of induction, \( P(t) \) is true for all positive integers \( t \).

**Lemma 4.** Let \( F: \mathbb{R}^n \to \mathbb{R}^n \), \( (n \geq 1) \) be continuous for a bounded closed region \( \Theta \subset \mathbb{R}^n \), and \( \Theta_0 = \{ \theta \in \Theta \mid F(\theta) = 0 \} \) has only a finite number of points. If a sequence \( \{\theta^k\}_{k=1}^{\infty} \subset \Theta \) satisfies

\[
\lim_{k \to \infty} \|\theta^{k+1} - \theta^k\| = 0, \quad \lim_{k \to \infty} \|F(\theta^k)\| = 0,
\]

then there exists a point \( \theta^* \in \Theta_0 \) such that

\[
\lim_{k \to \infty} \theta^k = \theta^*.
\]

**Proof.** The proof follows directly from Wu, Shao, and Qu (2005).

By virtue of lemmas 1 to 4, we can finally prove the following convergence theorem of the split-complex nonlinear gradient training procedure.

**Proof of Theorem 1 for SCNGN.** (a) Using the Taylor expansion and noting equation 2.3, equations 4.2 to 4.5, learning rule 4.7–4.8, and lemma 3, we have

\[
E(w^{k+1}) - E(w^k) = -\sum_{t=1}^{Q} (\text{ri}[d(t)] - \text{ri}[y^k_1(t)])^T \otimes e_1^T \text{ri}[\Delta y^k(t)]
\]

\[
+ \frac{1}{2} \sum_{t=1}^{Q} \|\Delta y^k_1(t)\|^2
\]

\[
= -\frac{1}{\eta} \|\Delta w^k\|^2 + \lambda^k,
\]

where

\[
\lambda^k = -\sum_{t=1}^{Q} (\text{ri}[d(t)] - \text{ri}[y^k_1(t)])^T \otimes e_1^T \delta^k(t) + \frac{1}{2} \sum_{t=1}^{Q} \|\Delta y^k_1(t)\|^2.
\]

It follows from assumptions A1 and A2, lemmas 2 and 3, and the Cauchy-Schwartz inequality that there exists a positive constant \( C \) such that

\[
|\lambda^k| \leq C \|\Delta w^k\|^2.
\]

(A.11)
Combining equations A.9 to A.11 yields

$$E(w^{k+1}) - E(w^k) \leq -\left(\frac{1}{\eta} - C\right) \|\Delta w^k\|^2. \quad (A.12)$$

Thus, the range of learning rate $\eta$ for convergence of the SCNGN algorithm is given by

$$0 < \eta < \frac{1}{C}. \quad (A.13)$$

(b) Set $\beta = \frac{1}{\eta} - C$. By equation A.12, we have

$$E(w^{K+1}) \leq E(w^K) - \beta \|\Delta w^K\|^2 \leq \cdots \leq E(w^0) - \beta \sum_{k=0}^{K} \|\Delta w^k\|^2. \quad (A.14)$$

Since $E(w^{K+1}) \geq 0$ for any $K \geq 0$, we let $K \to \infty$ to get

$$\sum_{k=0}^{\infty} \|\Delta w^k\|^2 \leq \frac{1}{\beta} E(w^0) < \infty, \quad (A.15)$$

and thus

$$\lim_{k \to \infty} \|\Delta w^k\|^2 = \lim_{k \to \infty} \|\Delta w^k\| = 0. \quad (A.16)$$

This, together with equation 4.8, leads to

$$\lim_{k \to \infty} \|\nabla_r E(w^k)\| = 0, \quad \lim_{k \to \infty} \|\nabla_i E(w^k)\| = 0. \quad (A.17)$$

(c) Take

$$\theta = ri[w], \quad F(\theta) = \begin{pmatrix} \nabla_r E(w) \\ \nabla_i E(w) \end{pmatrix}. \quad (A.18)$$

By equations A.16 and A.17 and assumption A3, we can see that $F(\theta)$ satisfies the conditions of lemma 4, so there exists a point $w^* \in \Phi_0$ such that

$$\lim_{k \to \infty} w^k = w^*. \quad (A.19)$$

This completes the proof of the strong convergence.
A.2 Convergence Analysis for SCNNGN. In place of lemma 3, we now have lemma:

**Lemma 5.** For the term $y^k_1(t)$ given in equation 5.4, there exists a second-order Taylor series expansion,

$$r_i \left[ \Delta y^k_1(t) \right] = \frac{\partial r_i[y^k_1(t)]}{\partial r_i[w]} r_i[\Delta w^k] + \alpha^k(t),$$  \hspace{1cm} (A.19)

where $\alpha^k(t) = I_2 \otimes e_1^T \delta^k(t)$, and $\delta^k(t)$ referred to equation A.8.

**Proof of Theorem 2 for SCNNGN.** (a) Using the Taylor expansion and noting equations 2.3, equations 4.2 to 4.5, the learning rule 5.1–5.2, and lemma 5, we have

$$E(w^{k+1}) - E(w^k) = - \sum_{t=1}^Q (r_i[d(t)] - r_i[y^k_1(t)])^T r_i[\Delta y^k_1(t)] + \frac{1}{2} \sum_{t=1}^Q \| \Delta y^k_1(t) \|^2$$

$$\leq - \frac{1}{\eta(k)} \| \Delta w^k \|^2 + \sum_{t=1}^Q \left\| \frac{\partial r_i[y^k_1(t)]}{\partial r_i[w]} \right\|^2 \| \Delta w^k \|^2 + \lambda^k,$$  \hspace{1cm} (A.20)

where

$$\lambda^k = - \sum_{t=1}^Q (r_i[d(t)] - r_i[y^k_1(t)])^T \alpha^k(t) + \sum_{t=1}^Q \| \alpha^k(t) \|^2.$$  \hspace{1cm} (A.21)

It follows from assumptions A1 and A2, lemmas 2 and 5, and the Cauchy-Schwartz inequality that there exists a positive constant $C$ such that

$$|\lambda^k| \leq C \| \Delta w^k \|^2.$$  \hspace{1cm} (A.22)

A combination of equations A.20 to A.22 leads to

$$E(w^{k+1}) - E(w^k) \leq - \frac{1}{\eta(k)} \| \Delta w^k \|^2 + \sum_{t=1}^Q \left\| \frac{\partial r_i[y^k_1(t)]}{\partial r_i[w]} \right\|^2 \| \Delta w^k \|^2$$

$$+ C \| \Delta w^k \|^2.$$  \hspace{1cm} (A.23)

Solving for $\eta(k)$, we can show that in order for the SCNNGN algorithm to converge, the following condition should be satisfied,

$$0 < \eta(k) < \frac{1}{\sum_{t=1}^Q \left\| \frac{\partial r_i[y^k_1(t)]}{\partial r_i[w]} \right\|^2 + C},$$  \hspace{1cm} (A.24)
which gives that the lower bound for convergence of SCNNGD with respect to $\varepsilon$ is

$$\varepsilon > \mu C.$$ \hfill (A.25)

We can use assumptions A1 to A3, lemmas 1 and 2, and lemmas 4 and 5 to obtain the weak and strong convergence results for SCNNGN precisely as in the proof of theorem 1 for SCNGN.

**A.3 Convergence Analysis for SCANNGN.** The convergence proof of SCANNGN is basically the same as that of SCNNGN.

**Proof of Theorem 2 for SCANNGN.** As in the proof of SCNNGN, we get a lower bound of the adaptive regularization parameter,

$$\varepsilon(k) > \mu C.$$ \hfill (A.26)

We can now use equation A.26 to determine the convergence boundaries of $\rho(k)$, using equation 6.10:

$$\varepsilon(0) - \mu \rho(k) \sum_{n=1}^{k} \frac{(\Psi(n))^{\top} \Psi(n - 1)}{\left(\sum_{i=1}^{Q} \left\| \frac{\partial r_i[y_{n-1}(t)]}{\partial r_i[w]} \right\|^2 + \varepsilon(n - 1)\right)^2} > \mu C.$$ \hfill (A.27)

Substituting the term for $\Psi(k)$, equation 6.4, we can write

$$\rho(k) \sum_{n=1}^{k} \frac{\eta(n - 1) (\Delta w^n)^{\top} \Delta w^{n-1}}{\eta(n)} < \mu (\varepsilon(0) - \mu^2 C).$$ \hfill (A.28)

Using the Cauchy-Schwarz inequality yields

$$\sum_{n=1}^{k} \frac{\eta(n - 1) (\Delta w^n)^{\top} \Delta w^{n-1}}{\eta(n)} < k \frac{\eta_{\text{max}}}{\eta_{\text{min}}} \max_{1 \leq n \leq k} \|\Delta w^n\|^2,$$ \hfill (A.29)

where $\eta_{\text{max}} = \max_{1 \leq n \leq k} \eta(n)$ and $\eta_{\text{min}} = \min_{1 \leq n \leq k} \eta(n)$. Thus, solving the inequality A.28 with respect to $\rho(k)$ gives

$$\rho(k) < \frac{1}{k} \frac{\eta_{\text{min}}}{\eta_{\text{max}}} \varepsilon(0) - \mu^2 C \max_{1 \leq n \leq k} \|\Delta w^n\|^2.$$ \hfill (A.30)
Acknowledgments

We thank Wei Wu and the anonymous reviewers for their insightful comments, which were helpful to us in revising this letter. This work is supported by the Fundamental Research Funds for the Central Universities (HEUCFR 1023).

References


Received November 20, 2009; accepted April 2, 2010.