Recurrent Neural Network Adaptive Equalizers Based on Data Communication

Hongrui Jiang and Kyung Sup Kwak

Abstract: **In this paper, a decision feedback recurrent neural network equalizer and a modified real time recurrent learning algorithm are proposed, and an adaptive adjusting of the learning step is also brought forward. Then, a complex case is considered. A decision feedback complex recurrent neural network equalizer and a modified complex real time recurrent learning algorithm are proposed. Moreover, weights of decision feedback recurrent neural network equalizer under burst-interference conditions are analyzed, and two anti-burst-interference algorithms to prevent equalizer from out of working are presented, which are applied to both real and complex cases. The performance of the recurrent neural network equalizer is analyzed based on numerical results.**

Index Terms: **Recurrent neural network, adaptive equalization, decision feedback, burst-interference, RTRL algorithm, CRTRL algorithm.**

I. INTRODUCTION

Channel equalization can be regarded as a mode classification problem. Since neural networks have good mode classification properties, different neural network structures are applied to channel adaptive equalization [1]–[3] with the development of neural networks technology. Various structures and algorithms of equalizers possess their own advantages and shortcomings. For example, provided enough freedom, the multilayer perceptron can be applied to arbitrary complicate non-linear channel equalization. But in project realization, there is always a contradiction between its properties and complexity of realization. The larger the structure, the longer the time needed for computing, and the smaller the data transmitting rate. Recurrent neural network (RNN) has the properties of small size and good performance, and it relieves the contradiction in the channel equalization [3]. Because it is similar to an IIR filter, it can get good equalization effects, or complete complicate non-linear map with only a few nodes.

In real case, recurrent neural network equalizer applying real time recurrent learning (RTRL) algorithm are presented in [3], [4]; in complex case, complex recurrent neural network equalizer (CRNNE) using complex real time recurrent learning (CRTRL) algorithm is first proposed in [5]. Up to now, no one has considered the anti-burst-interference problem for recurrent neural network equalizer. The purpose of this paper is to improve the performance of recurrent neural network equalizer from all possible aspects.

Manuscript received November 20, 2001; approved for publication by Gi-Hong Im, Division I Editor, December 26, 2002.

The authors are with the Graduate School of Information Technology and Telecommunications, Inha University, Korea, E-mail: jianghongrui@yahoo.com, kskwak@inha.ac.kr.

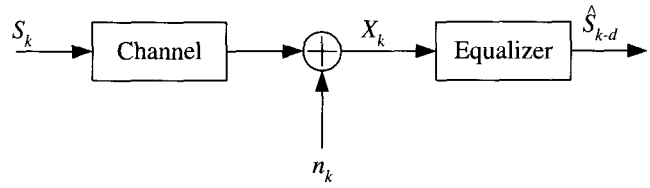


Fig. 1. Channel equalizer model.

The paper is organized as follows. Decision feedback recurrent neural network equalizer (DFRNNE) structure is given in Section II. Then modified real time recurrent learning (modified RTRL) algorithm is proposed, and adaptive adjusting of the learning step is also brought forward in Section III. Then, we will study complex case. Model of decision feedback complex recurrent neural network equalizer (DFCRNNE) is provided in Section IV. Modified complex real time recurrent learning (modified CRTRL) algorithm is proposed in Section V. Later on, the variance of weights for decision feedback recurrent neural network equalizer is studied for solving burst-interference problem in Section VI. Then two anti-burst-interference algorithms are given, which are applied to both real and complex cases in Section VII. Finally, conclusions are given in Section VIII.

II. DFRNNE STRUCTURES

The channel equalizer model is shown in Fig. 1 where s_k, n_k, x_k, and $\hat{s}_{k-d}$ represent signal, noise, input signal of equalizer and estimate signal, respectively.

In this section, the following distortion channel models [6] are used:

$$\text{LCH}: \quad x_k = 0.3482s_k + 0.8704s_{k-1} + 0.482s_{k-2}, \tag{1}$$

$$\text{NLCH}: \quad y_k = x_k + 0.2x_k^2, \tag{2}$$

where linear distortion channel with severe ISI (LCH) and nonlinear distortion channel (NLCH) are investigated. In the following simulative experiments, s_k is a random sequence with equal probability of -1 or +1 with unity power, and Gaussion white noise is added to the equalizer's reception end.

Fig. 2 is RNN structure with 3 nodes, where all neurons connect with each other, each input will be imported into each neuron, and each neuron output may be regarded as the external output of the network. x_k is the input signal of equalizer, and W_{ij} represents the connection weight which will be introduced in the following.

Fig. 3 is the DFRNNE structure (DFRNNE). We define that n, m, and l are the numbers of inner nodes, the delaying inputs and

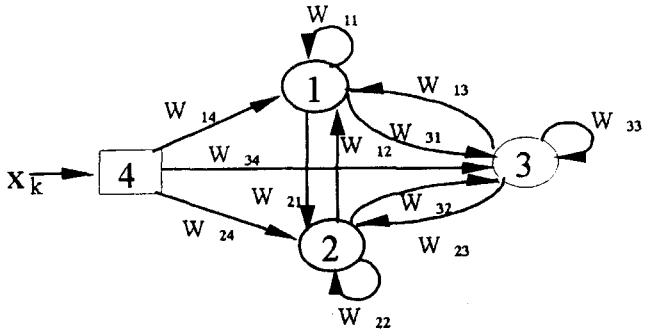


Fig. 2. RNN Structure with 3nodes.

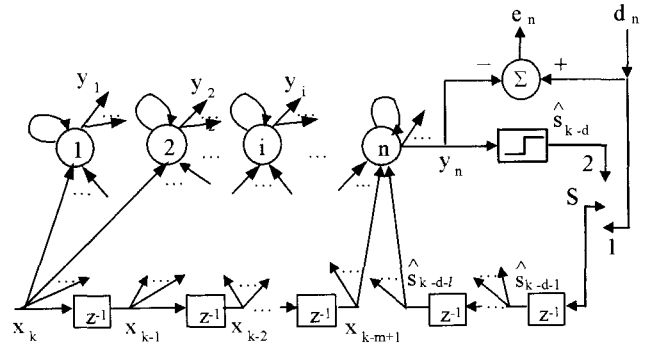


Fig. 3. Structure of DFRNNE.

the feedback delaying inputs of DFRNNE respectively. d_n is the training signal, e_n is the error of the nth neuron, $X_{k-i+1}(i = 1, 2, \cdots, n)$ is the delaying input signal, y_i is the output of the $ith(i = 1, 2, \cdots, n)$ node, and $\hat{s}_{k-d-i}(i = 1, 2, \cdots, l)$ is the feedback delayed input signal where d is the channel delay. In training process (switch S points to 1), training signal is regarded as the delaying input of each decision feedback signal so that effective information can be fully used and false propagation can be prevented. When signals propagate (switch S points to 2), equalizer's decision output becomes the delaying feedback input.

The weighted sum of the pth node's inputs is

$$v_p(t+1) = \sum_{i=1}^{n} w_{pi} y_i(t) + \sum_{i=1}^{m} w_{p,n+1} x_{k-i+1}(t) + \sum_{i=1}^{l} w_{p,n+i} \hat{s}_{k-d-i}, p = 1, 2, \cdots, n, \tag{3}$$

where w_{pi} is the weight from the ith $(i = 1, 2, \cdots, n)$ node to the pth node, w_{pj} is the weight from each delaying input signal $(j = n+1, \cdots, n+m)$ to the pth node, and w_{ph} is the weight from each decision feedback delaying signal $(h = n+m+1, \cdots, n+m+l)$ to the pth node.

Let the output of the ith $(i = 1, 2, \cdots, n)$ node

$$y_i(t+1) = f[v_i(t+1)], i = 1, 2, \cdots, n, \tag{4}$$

where the active function

$$f(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}. \tag{5}$$

Then we get the decision output

$$\hat{s}_{k-d} = SGN(y_n(t+1)). \tag{6}$$

III. LEARNING ALGORITHMS OF DFRNNE

A. Modified RTRL Algorithms

Real-time recurrent learning (RTRL) algorithm is applied to RNNE. Here, modified real time recurrent learning (modified RTRL) algorithm is proposed for DFRNNE.

$d_h(h = 1, 2, \cdots, n)$ is the training signal, and the error of the hth neuron is defined as

$$e_h(t+1) = d_h(t+1) - y_h(t+1), h = 1, 2, \cdots, n. \tag{7}$$

The networks instantaneous total error is given by

$$J(t+1) = \frac{1}{2} \sum_{h=1}^{n} e_h^2(t+1). \tag{8}$$

The objective of the algorithm updating the connecting weight w_{ij} is to minimize $J(t+1)$.

Define the kronecker delta

$$\delta_{ip} = \begin{cases} 1 & i = p \\ 0 & i \neq p \end{cases}, \tag{9}$$

and the derivative of the active function

$$f'(x) = \frac{4}{[\exp(x) + \exp(-x)]^2}. \tag{10}$$

The sensitivity is defined as

$$P_{ij}^{p}(t+1) = f'(v_p(t+1)) \left[\sum_{h=1}^{n} w_{ph}(t) p_{ij}^{h}(t) + \delta_{ip} z_i(t) \right], \tag{11}$$

where $p = 1, 2, \cdots, n$, and $z_j(t)$ represents the output of DFRNNE inner nodes $(j = 1, 2, \cdots, n)$, the external input signal $(j = n+1, \cdots, n+m)$ and feedback input signal $(j = n+m+1, \cdots, n+m+l)$.

$$w_{ij}(t+1) = w_{ij}(t) + \alpha \sum_{h=1}^{n} e_h(t+1) p_{ij}^{h}(t+1) + u(w_{ij}(t) - w_{ij}(t-1)), \tag{12}$$

where α is the learning step of the adaptive equalizer. $\alpha > 0, i = 1, 2, \cdots, n, j = 1, 2, \cdots, n, n+1, \cdots, n+m, n+m+1, \cdots, n+m+l$. Moreover, momentum factor u $(0 < u < 0.001)$ is introduced to help update the weights.

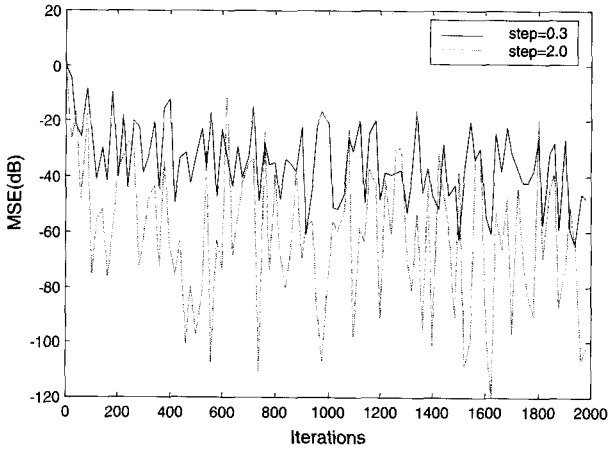


Fig. 4. The RNNE learning curves with different α (LCH, SNR=22dB).

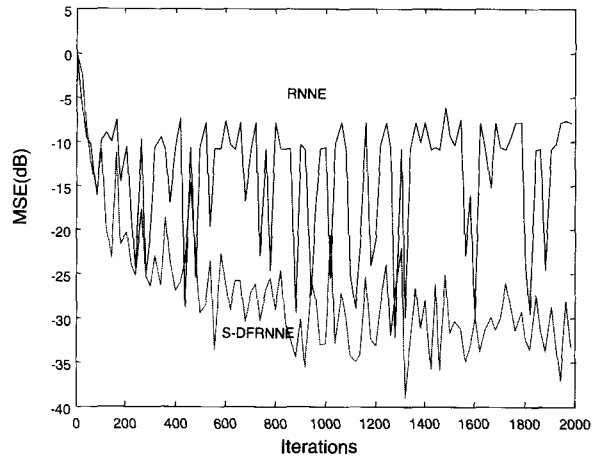


Fig. 5. Learning curves(LCH).

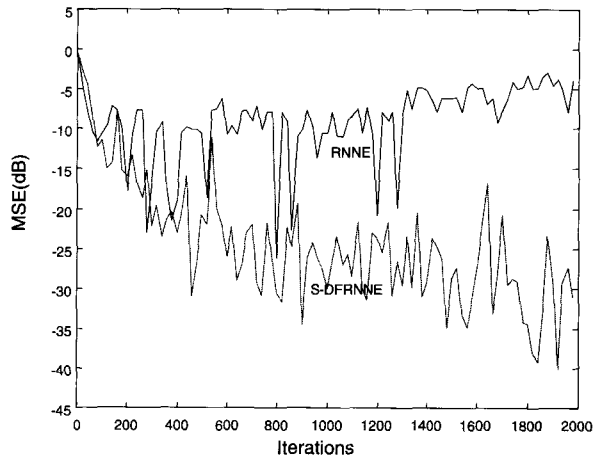


Fig. 6. Learning curves(LCH+NLCH).

B. Adaptive Adjusting of the Learning Step

The learning step α has a certain effect on the convergency speed of algorithm. Here, we use the channel LCH which is presented in section II. Fig. 4 is RNNE's learning curves with different α where MSE and SNR represent mean square error and signal-to-noise ratio respectively. Here, SNR=22dB. As shown in Fig. 4, within certain range of values, the convergency speed of algorithm increases with the increase of α. For larger α, the algorithm converges quickly, but is subject to vibration and instability; For smaller α, the algorithm converges slowly, and is subject to trapping into local minimization. So the selection of α is very important. The methods of adaptive adjusting α are proposed according to the above analysis. Its basic idea is to continuously adjust α after each iteration, which possesses two objectives: one is to let the algorithm skip out of the local minimization and speed up the convergence process, the other is to try to avoid the instability of algorithm.

Firstly, define the system's total error E_n which is equal to $\Sigma J(t+1)$. Use an exponential function to realize the adaptive adjusting of α. This function regards E_n as an independent variable.

Set

$$\alpha = \alpha_0 \exp(E_n), 0.1 < \alpha_0 < 1.5. \tag{13}$$

The algorithm will adaptively adjust α during the iteration processes. If the total error is large, α will decrease; if the total error is small, α will increase. The total error will become small with increased iterations, and α will gradually hold a certain level.

C. Simulation Study

In accordance with DFRNNE, the simulative experiments are conducted for the comparison with the traditional RNNE under the same condition. Here, we use the channels presented in section II.

C.1 Comparison of Learning Performance

In every experiment, initial weights w_{ij}, initial sensitivity P_{ij} and initial output y_i of each node are random numbers whose absolute values are less than or equal to 10^{-3}. The parameters of traditional RNNE and DFRNNE are $m = 2, n = 1, \alpha = 0.5$ and $m = 2, n = 1, l = 3, \alpha_0 = 0.5$. Therefore, two groups of learning curves are obtained in accordance with two different distortion channels respectively.

In Fig. 5, the distortion channel is LCH, SNR=16dB. In Fig. 6, the distortion channel is LCH cascade-connected with NLCH, SNR =18dB. According to Fig. 5, S-DFRNNE converges at about -30dB, while RNNE at about -10dB with greater vibration. According to Fig. 6, S-DFRNNE converges at about -30dB, while RNNE doesn't converge at all. In short, S-DFRNNE has better learning performance than the traditional RNNE.

C.2 Comparison of Bit Error Rate(BER)

In order to research the BER performance of DFRNNE, two groups of BER curves (Fig. 7 and 8) are obtained according to the above distortion channels.

According to these curves, we note that DFRNNE has better BER performance than RNNE. During the simulation process, we find that, for the two given distortion channels, the BER of RNNE always presents inconsistency and possesses certain vi-

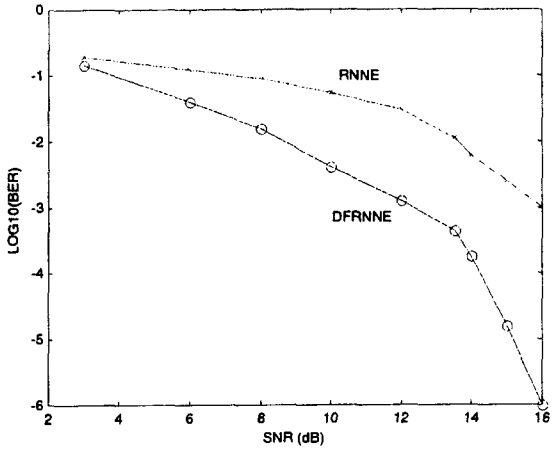


Fig. 7. BER curves (LCH).

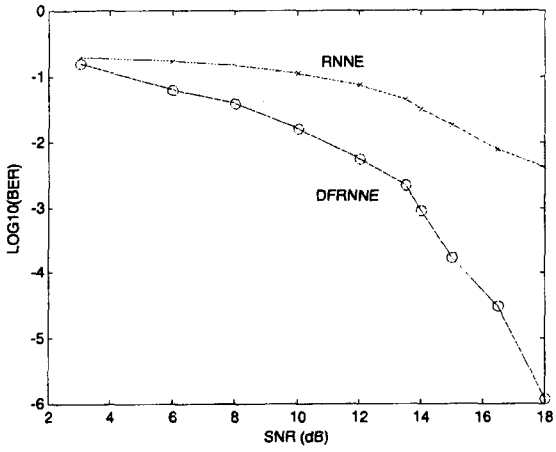


Fig. 8. BER curves (LCH+NLCH).

bration, and that even the increase of SNR deteriorates the BER performances. The fault lies on applying RNNE to the channel equalization, while DFRNNE makes up for the fault and its BER possesses consistency.

IV. MODEL OF DFCRNNE

We design a better performing DFRNNE based on the real case studied before. However, when a QAM signal with the complicated channels is considered, neural network equalizers based on real analysis have a large limitation. Some researchers extend recurrent neural network equalizer based on RTRL algorithm to complex recurrent neural network equalizer (CRNNE) based on complex real-time recurrent learning (CRTRL) algorithm [5] whose inputs, outputs, weights and active functions are all complex.

Decision feedback complex recurrent neural network equalizer (DFCRNNE) is presented, and modified CRTRL algorithm, which fits the structure, is deduced in the following. DFCRNNE not only inherits the advantages of CRNNE on complex field, but also skillfully puts the traditional decision feedback structure for linear channels equalization into RNN, and replaces decision feedback signals with training signals in the learning process. We will present the structure of DFCRNNE and modified

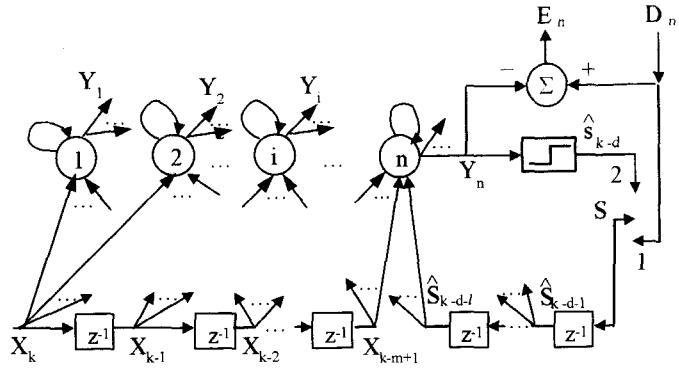


Fig. 9. The structure of DFCRNNE.

CTRL algorithm in the following.

Lower case (upper case) is used to represent real (complex) in this paper. Structure of DFCRNNE is shown in Fig. 9. In Fig. 9, we define that n is the inner nodes number of the DFCRNNE, m the delaying input number of the equalizer, l the feedback delaying input number of the equalizer,$X_{k-i+1}(i = 1, 2, \cdots, m)$ is the delaying input signal, $Y_k(k = 1, 2, \cdots, n)$ is the output of the kth node, D_k is the training signal of the kth node, E_n is the error of the nth neuron, and $\hat{S}_{k-d-t}(t = 1, 2, \cdots, l)$ is the feedback input signal with delaying t. Modified CRTRL algorithm is adopted for adjusting the equalizers weights. In training process (switch S points to 1), training signal is regarded as the delaying input of each decision feedback signal so that effective information can be taken into full use and false propagation can be prevented. In data transmission process (switch S points to 2), equalizer's decision output becomes the delaying feedback input.

V. MODIFIED CRTRL ALGORITHM AND SIMULATION

A. Modified CRTRL Algorithm

CRTRL algorithm is used for the training process in CRRNE [5]. Here, modified CRTRL algorithm is proposed and used for the training process in DFCRRNE.

Denote V_k as the weighted sum of the kth $(k = 1, 2, \cdots, n)$ node's inputs. W_{ij} represents the weight from the jth $(j = 1, 2, \cdots, n)$ node to the ith node, the weight from each delaying input signal $(j = n + 1, \cdots, n + m)$ to the ith node, or the weight from each feedback delaying signal $(j = n + m + 1, \cdots, n+m+l)$ to the ith node. We set real part and imaginary part of W as w_R and w_I. Weights, exterior inputs, feedback inputs, expected response and the weighted sum of the kth node's inputs are all complex.

Define the following activation function [4],

$$F(Z) = f(z_R) + jf(z_I). \tag{14}$$

In our study, we select

$$f(z) = \frac{1 - \exp(-2z)}{1 + \exp(-2z)}, \tag{15}$$

and the differential of f(z) is

$$f'(z) = \frac{4}{[\exp(z) + \exp(-z)]^2}. \tag{16}$$

To calculate the steepest descent direction, we use the concept of the gradient instead of the complex derivative. If c_R and c_I are the unit vectors in the direction of z_R and z_I respectively, we get

$$\nabla_{(z)}F(Z) = \frac{\partial F(Z)}{\partial z_R}C_R + \frac{\partial F(Z)}{\partial z_I}c_I. \tag{17}$$

If C_R=1 and C_I ($j = \sqrt{-1}$), then

$$\nabla_z F(Z) = \frac{\partial F(Z)}{\partial z_R} + \frac{\partial F(Z)}{\partial z_I}j. \tag{18}$$

The weighted sum of the kth ($k = 1, 2, \cdots, n$) node's is

$$\begin{aligned} V_k[t+1] = &\sum_{f=1}^{n} W_{kf}Y_f[t] + \sum_{f=1}^{m} W_{k,n+f}X_{k-f+1}[t] \\ &+ \sum_{f=1}^{l} W_{k,n+m+f}\hat{S}_{k-d-f}. \end{aligned} \tag{19}$$

The output of the kth node is

$$Y_k(t+1) = F(V_k[k+1]) = f(v_{RK}[t+1]) + jf(v_{Ik}[t+1]). \tag{20}$$

Define $E_k(t)$ as the complex error of the $k$the neural node.

$$E_k(t) = D_k(t) - Y_k(t) = e_{Rk}[t] + je_{Ik}[t]. \tag{21}$$

The networks instantaneous total error is

$$J[t+1] = \frac{1}{2}\sum_{k=1}^{n}|E_k(t+1)|^2. \tag{22}$$

The objective of the algorithm updating the connecting weights $W_{ij}(i = 1, 2, \cdots, n, j = 1, 2, \cdots, n, n+1, \cdots, n+m, n+m+1, \cdots, n+m+l)$ is to minimize $J[t+1]$. We introduce the sensitivity terms P_{RR}, P_{RI}, P_{IR}, and P_{II} [5] defined by

$$\begin{bmatrix} P^k_{PRij} & P^k_{PIij} \\ P^k_{IRij} & P^k_{PIIj} \end{bmatrix}[t] = def \begin{bmatrix} \frac{\partial y_{RK}}{\partial w_{Rij}} & \frac{\partial y_{RK}}{\partial w_{Iij}} \\ \frac{\partial y_{Ik}}{\partial w_{Rij}} & \frac{\partial y_{Ik}}{\partial w_{Iij}} \end{bmatrix}[t]. \tag{23}$$

Using (18), we compute the gradient of $J[t]$ with respect to W_{ij}.

$$\nabla_{w_{ij}}J[t] = \frac{\partial J[t]}{\partial w_{Rij}} + j\frac{\partial J[t]}{\partial w_{Iij}}. \tag{24}$$

Then, differentiating (22), we get

$$\frac{\partial J[t]}{\partial w_{Rij}} = -\sum_{k=1}^{n} e_{Rk}[t]\frac{\partial y_{Rk}[t]}{\partial w_{Rij}} - \sum_{k=1}^{n} e_{Ik}[t]\frac{\partial y_{Ik}[t]}{\partial w_{Rij}}, \tag{25}$$

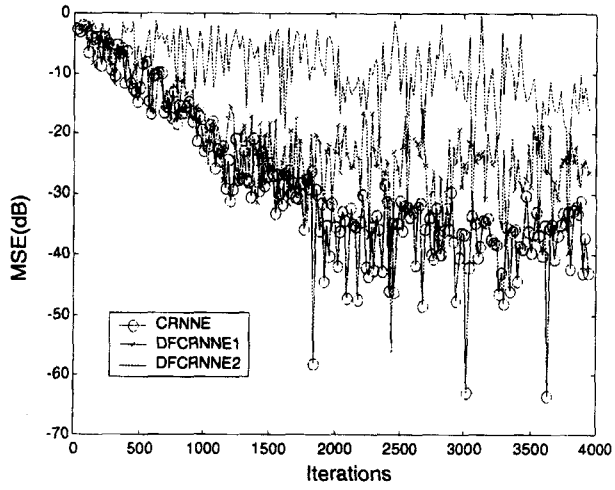


Fig. 10. Comparison of MSE curves of CRNNE and DFCRNNE under linear channel.

$$\frac{\partial J[t]}{\partial w_{Iij}} = -\sum_{k=1}^{n} e_{Rk}[t]\frac{\partial y_{Rk}[t]}{\partial w_{Iij}} - \sum_{k=1}^{n} e_{Ik}[t]\frac{\partial y_{Ik}[t]}{\partial w_{Iij}}, \tag{26}$$

In order to find a recursive relation for the calculation of the terms P_{RR}, P_{RI}, P_{IR}, and P_{II}, we differentiate (20) and get

$$\frac{\partial y_{Rk}[t+1]}{\partial w_{Rij}} = f'(v_{Rk}[t+1])\frac{\partial v_{Rk}[t+1]}{\partial w_{Rij}}. \tag{27}$$

Using weight W_{ij}, node output $Y_f(f = 1, \cdots, n)$, delaying input $X_{k-f+1}(f = 1, \cdots, m)$ and real part and imaginary part of feedback delaying input $\hat{S}_{k-d-f}(f = 1, \cdots, l)$ to rewrite (19), we get

$$\begin{aligned} v_{Rk}[t+1] = &\sum_{f=1}^{n}(w_{Rkf}y_{Rf}[t] - w_{Ikf}y_{If}[t]) \\ &+ \sum_{f=1}^{m} w_{Rk,n+f}x_{R,k-f+1}[t] - w_{Ik,n+f}x_{I,k-f+1}[t] \\ &+ \sum_{f=1}^{t}(w_{Rk,n+m+f}\hat{s}_{R,k-d-f} - w_{Ik,n+m+f}\hat{s}_{I,k-d-f}). \end{aligned} \tag{28}$$

After we compute the derivative of (28) with respect to w_{Rij}, in accordance with (23) and (27), we can get

$$\begin{aligned} &P^k_{RRij}(t+1) = f'(v_{Rk}[t+1]) \\ &\left[\sum_{p=1}^{n}(w_{Rkp}[t]P^p_{RRij}[t] - w_{Ikp}P^p_{IRij}[t]) + \partial_{ik}z_{Rj}[t]\right], \end{aligned} \tag{29}$$

where $i = 1, 2, \cdots, n, k = 1, 2, \cdots, n, j = 1, 2, \cdots, n, n+1, \cdots, n+m, n+m+1, \cdots, n+m+l$, and

$$\delta_{ik} = \begin{cases} 1 & i = k \\ 0 & i \neq k \end{cases}, \tag{30}$$

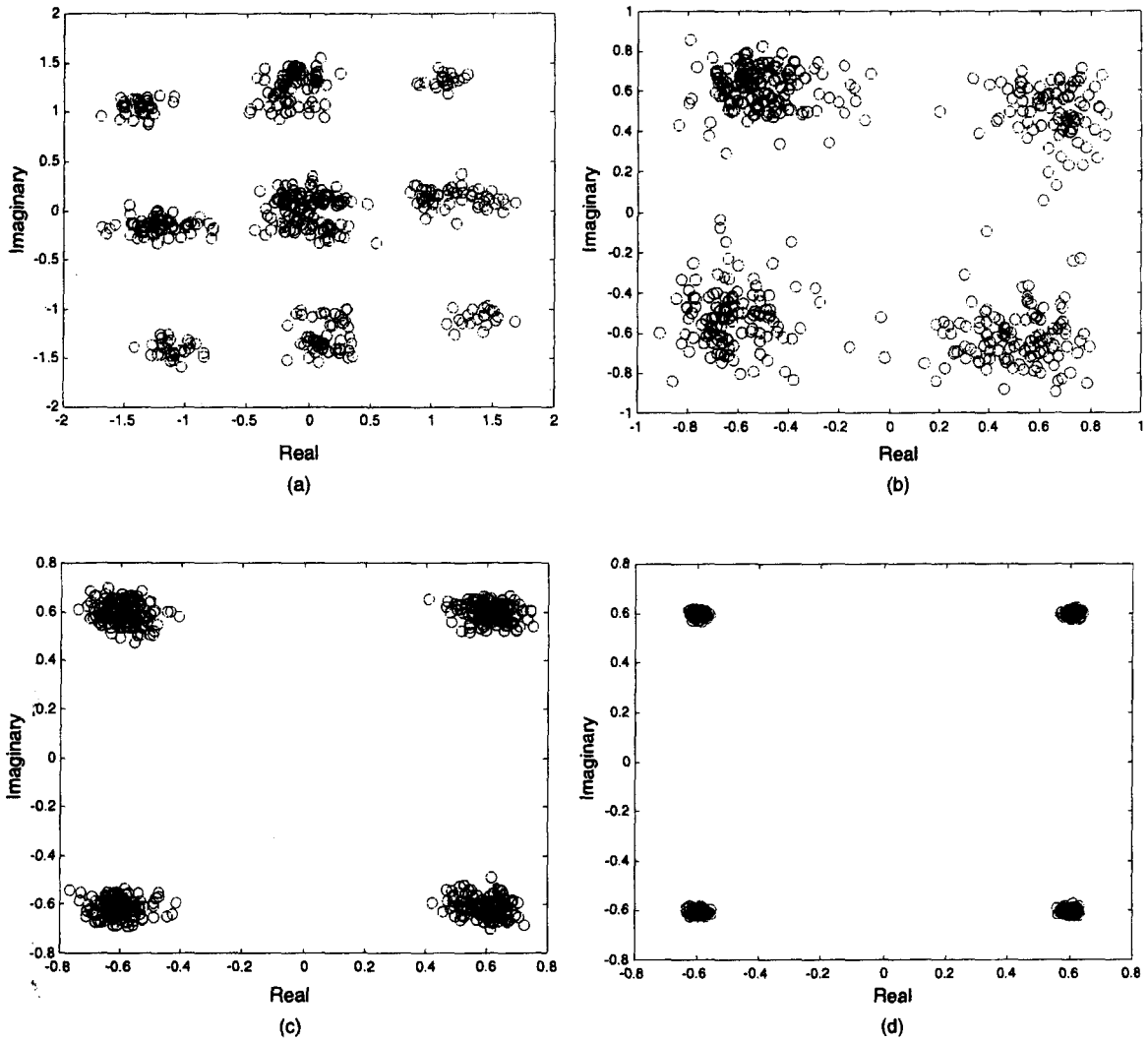


Fig. 11. Signal constellation plots before and after equalizaton under linear channel: (a) Before equalization, (b) CRNNE after equalization, (c) DFCRNNE1 after equalization (d) DFCRNNE2 after equalization.

and $Z_{Rj}[t]$ may represent the real part of DFCRNNE's inner node output ($j = 1, 2, \cdots, n$), input signal ($j = n+1, \cdots, n+m$), or feedback input signal ($j = n+m+1, \cdots, n+m+l$). In a similar manner $Z_{Ij}[t]$, in the following represents the imaginary part.

We may derive recursive equations for P_{RI}, P_{IR}, and P_{II} in a similar fashion. We group these recursive equations together to get the following recursive matrix equations for the sensitivity terms at the bottom of this page.

Finally, we get the weight update equation at the bottom of next page.

where α is the learning step of the adaptive process.

B. Simulation

To testify the equalization performance of DFCRNNE, we compare DFCRNNE with the traditional CRNNE [5] under the same conditions. For CRNNE, we set $n = 3, m = 1$, and for DFCRNNE, we set $n = 3, m = 1, l = 1$, and 2 respectively.

The input sequence $\{S_k\}$ is complex 4-QAM whose real and imaginary parts are assumed the values +0.6 or -0.6, and 15dB for SNR. In simulation, we first train 4000 times, then adopt decision guidance method to test equalization performance in process of data transmission. Small random complex values with $|W_{ij}(0)| < 10^{-4}$ are used to initialize the weights. Initial val-

$$\begin{bmatrix} P^k_{RRij} & P^k_{RIij} \\ P^k_{IRij} & P^k_{IIij} \end{bmatrix}[t] = \begin{bmatrix} f'(u_{Rk}) & 0 \\ 0 & f'(u_{ik}) \end{bmatrix} \times \left(\sum_{p=1}^{n} \begin{bmatrix} w_{Rkp} & -w_{Ikp} \\ w_{Ikp} & w_{Rkp} \end{bmatrix} \begin{bmatrix} P^p_{RRij} & P^p_{RIij} \\ P^p_{IRij} & P^p_{IIij} \end{bmatrix}[t] + \partial_{ik} \begin{bmatrix} z_{Rj} & -zIj \\ z_{IJ} & z_{Rj} \end{bmatrix}[t] \right). \tag{31}$$

ues of all parts of sensitivity term P are set to 0, and the learning step is fixed to $\alpha = 0.01$.

Case (1): We use the following linear channel, and channel output is

$$\begin{aligned} X_k &= S_k + (0.2501 + i^*1.0246)S_{k-1} \\ &\quad + (-0.0686 + i^*0.8237)S_{k-2} \\ &\quad + (-0.6819) + i^*0.0840)S_{k-3} + N_k. \end{aligned} \tag{33}$$

where noise N_k is complex white Gaussian noise whose imaginary part is the Hilbert transformation of real part.

Comparison of mean square error (MSE) curves of CRNNE and DFCRNNE under linear channel is given in Fig. 10 where DFCRNNE1 and DFCRNNE2 represent DFCRNNE with l=1 and 2 respectively.

From Fig. 10, we learn that convergence performance of DFCRNNE2 is better than that of DFRNNE1, and convergence performance of DFCRNNE1 is better than that of CRNNE.

Signal constellation plots before and after equalization under linear channel are shown in Fig. 11. In Fig. 11(b), there are some spots near the decision boundary. From Fig. 11(c) and Fig. 11(d), we see that DFCRNNE1 and DFCRNNE2 have better pattern classification characters compared to CRNNE, and no spots occur near the decision boundary, and DFCRNNE2 has better conglomerate performance than DFCRNNE1.

Case (2): We simulate a highly nonlinear communication channel. The transmitted signals $\{S_k\}$ are first passed through the linear channel in case 1 and then raised to higher powers. The output of channel can be expressed as:

$$X_k = (H^*S)_k + \sum_{i=2}^{5} l_i((H^*S)k)^i + N_k, \tag{34}$$

where H is the impulse response of the linear channel in case 1 and * represents convolution. The coefficient $l_i (i = 2, \cdots, 5)$ determines the amount of nonlinear harmonics added to the linear response. In the example, the values of the coefficients l_i are: $l_2 = 0.15, l_3 = 0.10, l_4 = 0.05$, and $l_5 = 0.10$.

Comparison of mean square error (MSE) curves of DFCRNNE and CRNNE under non-linear channel is given in Fig. 12. From Fig. 12, we see that both DFCRNNE1 and DFCRNNE2 have better convergence performances than CRNNE under non-linear channel.

Signal constellation plots before and after equalization under non-linear channel are shown in Fig. 13. There are many spots near the decision boundary for CRNNE in Fig. 13(b). It means CRNNE cannot get accurate estimation. From Fig. 13(c), we see DFCRNNE1 has better pattern classification capability than CRNNE, but there are still some spots near the decision boundary. From Fig. 13(d), we see DFCRNNE2 has the best conglomerate performance, and the data can be decided more accurately.

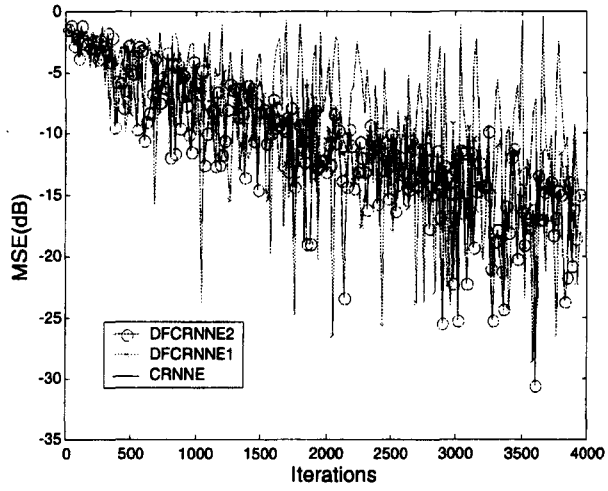


Fig. 12. Comparison of MSE curves of CRNNE and DFCRNNE under non-linear channel.

VI. ANALYSIS OF DFRNNE

With the development of neural networks technology, neural network adaptive equalization is extensively used to HF channel communications and its excellent effects are obtained. From above, we know DFRNNE has a simple structure, good stability and convergence, and low bit error rate [6]. Generally, channels' signal-to-noise ratio (SNR) is between 10dB and 20dB. However, since there exists all kinds of effects of uncertain factors such as burst-interference, it is fairly possible for channels to have SNR less than 10dB [7]. Simulation results have shown that when channels are affected by burst-interference, which makes SNR keep lower than 10dB for a long time, equalization performances of DFRNNE become bad or even inefficient. It is very possible to lose tracking ability and unable to recover the normal work when channels regain the normal condition, which will lead to interruption of communication under serious conditions. The whole system must be restarted at that time, which will affect the efficiency of communication. Therefore, it is necessary to improve performances of DFRNNE and enable it to adaptively track the dynamic variation of channels.

Based on the structure of DFRNNE we use, the channels presented in the Section II and LCH is cascade-connected with NLCH. SNR is initially set to be 16dB, and the initial values may be referred to Section III. When DFRNNE is used in equalization without burst-interference, it can always come to convergence condition. After training process ends, data transmission process subsequently begins. The variability of equalizer's weight values and mean-square error (MSE) are explored as follows.

We assume, without loss of generality, input number m=1, neuron number n=2, and feedback input number l=1. In fact, this is the structure of S-DFRNNE with 2 nodes, which is shown

$$W_{ij} = W_{ij} + \alpha\nabla_{w_{ij}}J(t) = W_{ij}(t) + \alpha\sum_{k=1}^{n}\left([e_{Rk}\ \ e_{ik}]\begin{bmatrix} P^k_{RRij} & P^k_{RIij} \\ P^k_{IRij} & P^k_{IIij} \end{bmatrix}\begin{bmatrix} 1 \\ j \end{bmatrix}(t)\right). \tag{32}$$

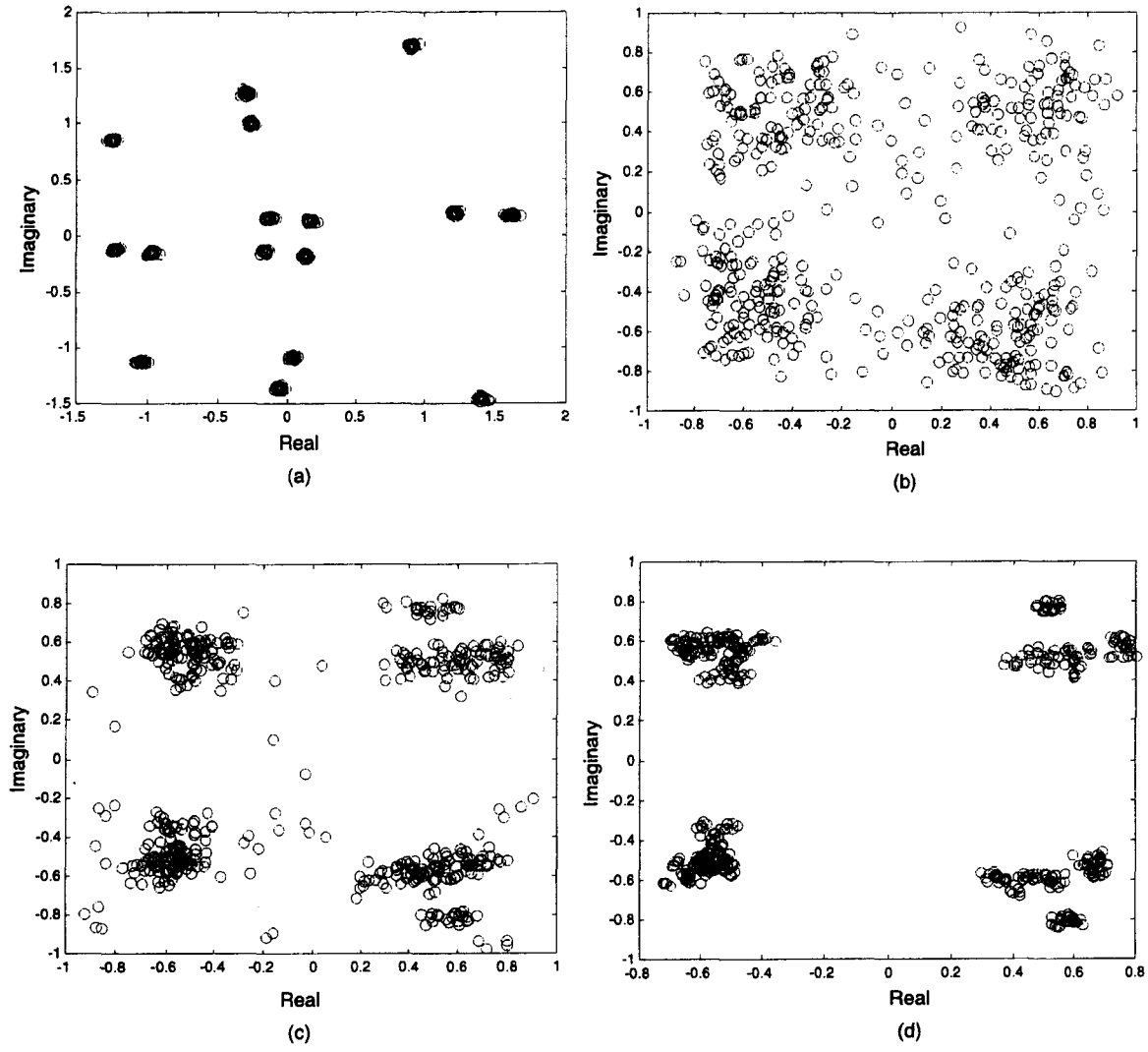


Fig. 13. Constellation plots before and after equalization under non-linear channel: (a) Before equalization, (b) CRNNE after equalization, (c) DFCRNNE1 after equalization, (d) DFCRNNE2 after equalization.

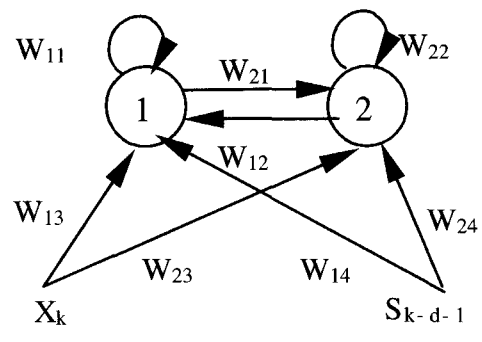


Fig. 14. The structure of DFRNNE with 2 nodes.

in Fig. 14. From Fig. 14, we consider eight weights: W_{11} (node1 $\rightarrow$ node1), W_{12} (node2 $\rightarrow$ node1), W_{13} (input $\rightarrow$ node1), W_{14} (feedback $\rightarrow$ input node1), W_{21} (node1 $\rightarrow$ node2), W_{22} (node2 $\rightarrow$ node2), W_{23} (input $\rightarrow$ node2), and W_{24} (feedback input $\rightarrow$ node2). Variation curves of eight weight values and MSE in normal channel are given in Fig. 15. According to Fig. 15(a), we know that every weight value curve basically keeps near a certain value to slightly fluctuate in the whole process of data transmission. Some of weights coincide: from up to down, two weights W_{13} and W_{23} overlap, four weights W_{11}, W_{12}, W_{21}, and W_{22} overlap, and two weights W_{14} and W_{24} overlap. MSE possesses very low values according to Fig. 15(b), and its bit error rate keeps at a very low level.

In data transmission process, when channel meets burst-interference during the period from the 300^{th} iteration to the 600^{th} iteration and SNR abruptly decreases by 3dB, DFRNNE cannot recover normal work condition after the channel resumes the normal condition. Communication must be interrupted so that the system must be restarted. In the burst-interference condition that SNR=3dB and interference duration is 300 iterations, variation curves of weight values and MSE with burst-interference are shown in Fig. 16.

From Fig. 16(a), it is known that every weight value curve declines very much when there is burst-interference, then con-

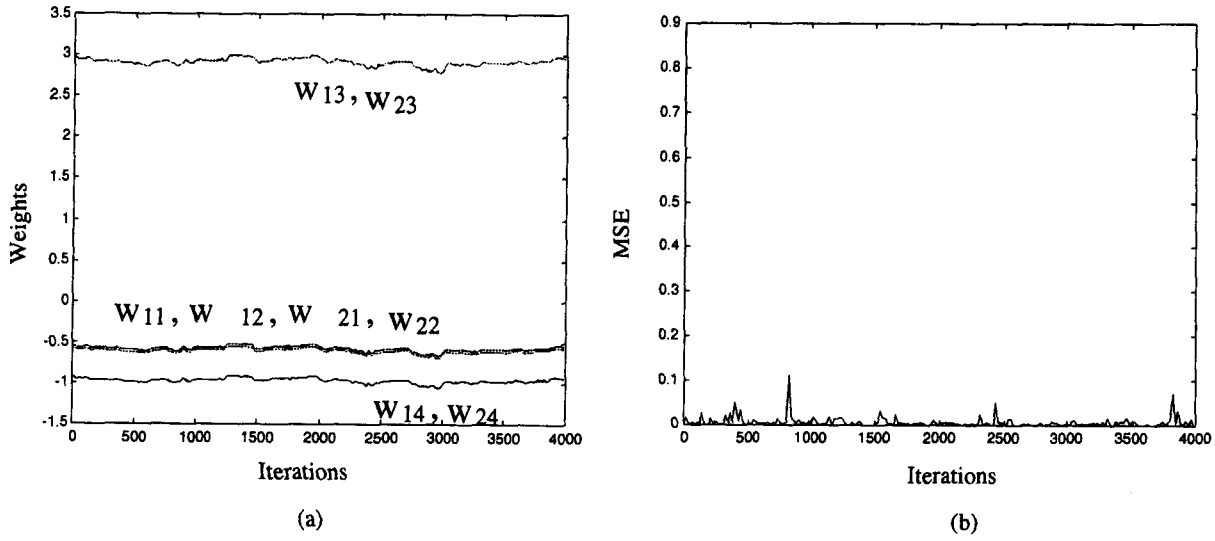


Fig. 15. Variation curves in normal channel: (a) Weight values, (b) MSE.

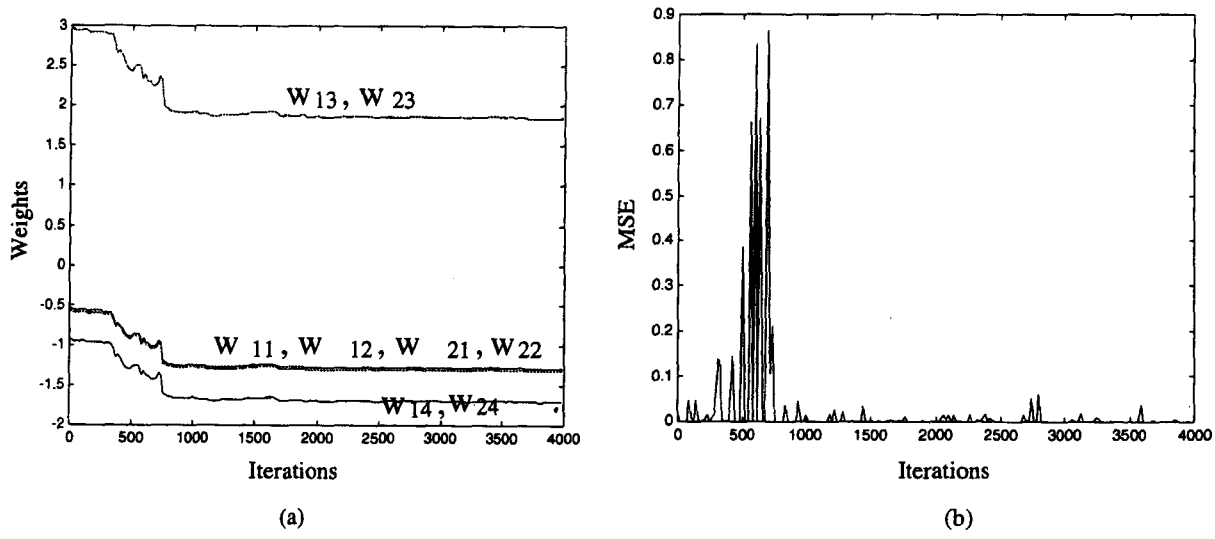


Fig. 16. Variation curves with burst-interference: (a) Weight values, (b) MSE.

verges to a certain value. From Fig. 16 (b), we know that MSE approximately converges to 0 after the equalizer has iterated 800 times. At that time, bit error rate is up to 0.4350, and equalizer keeps convergence condition. In fact, it is false convergence because the bit error rate is so large. Weight values deviate from the stable value with two conditions: divergence and false convergence. The latter often occurs in simulation. The fact that the weight values converge to a certain value must lead MSE to converge to 0, which may be determined by algorithm. At that time, equalizer loses efficiency, and the uncertainty of estimated bits become very large such that the bit error rate rapidly increases.

$$F_1[w_{ij}t] = \begin{cases} w_{ij}^{opt} & when \quad t > t_0 \\ w_{ij}(t) & when \quad t \le t_0 \end{cases}, \qquad (35)$$

VII. ANTI-BURST-INTERFERENCE ALGORITHMS

From the previous analysis, the main reason of the failure for S-DFRNNE is that weight $w_{ij}(i = 1, \cdots, n; j = 1, \cdots, n, n+1, \cdots, n+m, n+m+1, \cdots, n+m+l)$ diverges or falsely converges, causing random decision results. So we need control weight w_{ij} . Since w_{ij} varies with time, set w_{ij} as $w_{ij}(t)$ at time t. Define t_0 as certain time after equalizer converges. It is obvious that $w_{ij}(t_o)$ is near the stable value. Assume that the stable value is w_{ij}^{opt} with $w_{ij}^{opt} = w_{ij}(t_o)$. Define the following control functions:

Where α_1 and α_2 are some positive real numbers. Two modified algorithms are presented in the following.

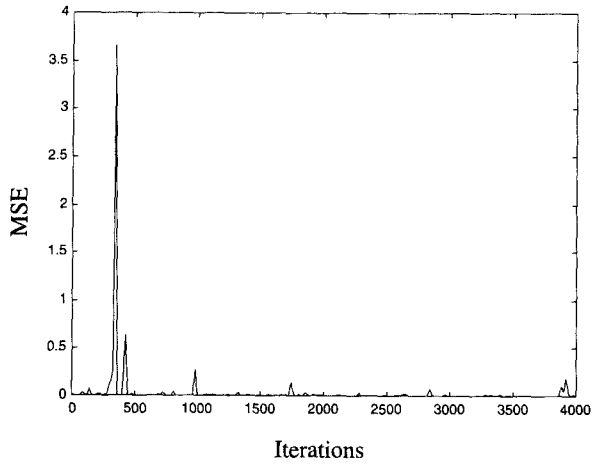


Fig. 17. The MSE of Anti-burst-interference algorithm 1 with burst-intererence.

A. *Anti-burst-interference Algorithm 1*

RTRL algorithm can still be used in the process of training. Then, set $w_{ij} = F_1[w_{ij}(t)]$. The evidences of the theory are that every weight value curve basically keeps near a certain value to slightly fluctuate in the whole process of data transmission when there is no burst-interference. Therefore, good equalization performances can be obtained by the stable value w_{ij} in the process of data transmission. The MSE of modified algorithm 1 with burst-interference (300 iterations) is given in Fig. 17. After channel recovers normal condition, the MSE is very small, and bit error rate is 0.008. Here, the MSE uses the output values (before decision and after decision) to compare. In this situation, capabilities of equalizer are good, and the equalizer is efficient to prevent false convergence conditions. However, because the fact that the weights keep invariant decreases the sensitivity to channel, the second algorithm is introduced.

B. *Anti-burst-interference Algorithm 2*

The second algorithm is given as follows. Again apply RTRL algorithm in the process of training, then set $w_{ij} = F_2[w_{ij}(t)]$. The second algorithm makes all kinds of weights be controlled within the effective range of its stable value. Suitable selection of α_1 and α_2 not only can prevent the weights of equalizer from deviating the stable value, but also unrestricts the equalizer in real channels in all cases, which amplifies ability and flexibility of DFRNNE.

When there is burst-interference (300 iterations), the variation curves of weights and MSE of the second modified algorithm are given in Fig. 18. When channel is normal, performance of DFRNNE does not change. When the channel meets burst-interference, MSE changes rapidly and bit error rate increases. When the channel becomes normal, equalizer recovers good working conditions after the transmission time of 50 iterations. From Fig. 18 (a), we know that every weight drives to a stable value, and some curves are slightly separated. From Fig. 18 (b), we know that MSE is always controlled within a very small range all the time. The bit error rate is 0.00001. We notice that α_1 and α_2 can be generally set as $0.2 \sim 0.3$ through a great deal of experiments.

In conclusion, the anti-burst-interference algorithms can effectively prevent false convergence when channel meets burst-interference, and make the equalizer automatically recover normal working conditions and continue to work.

C. *Applications on Complex Recurrent Neural Network Adaptive Equalization*

Two anti-burst-interference algorithms based on S-DFRNNE are proposed above. Theoretically, the two algorithms can be applied to all structures of recurrent neural network equalization. Here, we will verify them in the complex case. DFCRNNE are adopted. In the process of data transmission, when channel meets burst-interference during the time from the 300th iteration to 750th iteration, SNR abruptly decreases to 3dB. After the 750th iteration, channel recovers to normal condition.

MSE and signal constellation plots without anti-burst-interference measure are shown in Fig. 19. Then, when there is burst-interference, MSE and signal constellation plots with anti-burst-interference algorithms 1 and 2 are shown in Fig. 20 and Fig. 21 respectively.

From Fig. 19, we know when no anti-burst-interference measure is present, MSE approximately keeps zero after the 1000th iteration. However, signal conglomerate performance is bad, after equalization, so the signal cannot correctly be decided. It can be concluded that convergence at this time is false convergence. From Fig. 20 and Fig. 21, we know, with anti-burst-interference measure, MSE nearly converges to 0 after the 750th iteration, and signal's classification capability and conglomerate performance are better so that the signal can correctly be decided after equalization. According to MSE and signal constellation plots, it is obvious that system performances are improved with anti-burst-interference measure.

VIII. CONCLUSION

RNNE is very sensitive to the setting of various initial values. However, DFRNNE is not sensitive to various initial values, and adaptive adjusting of the learning step is brought forward in DFRNNE. In adaptive training, it is only needed to set any initial value as smaller random data and properly select adaptive step, which will gain better equalization effects that are sufficiently supported by the consistency of the learning and BER curves. DFRNNE has better and more stable equalization performance than RNNE.

DFCRNNE has better equalization properties than CRNNE. DFCRNNE classifies system patterns in the field of complex, which not only effectively compensates a variety of complicated

$$F_2[w_{ij}(t)] = \begin{cases} w_{ij}^{opt}[when \;\; t > t_0 \;\; and \;\; (w_{ij}(t) - w_{ij}^{opt} < -\alpha_1, \;\; or \;\; w_{ij}(t) - w_{ij}^{opt} > \alpha_2)] \\ w_{ij}(t)[when \;\; t \le t_0 \;\; or \;\; -\alpha_1 \le w_{ij}(t) - w_{ij}^{opt} \le \alpha_2] \end{cases} . \quad (36)$$

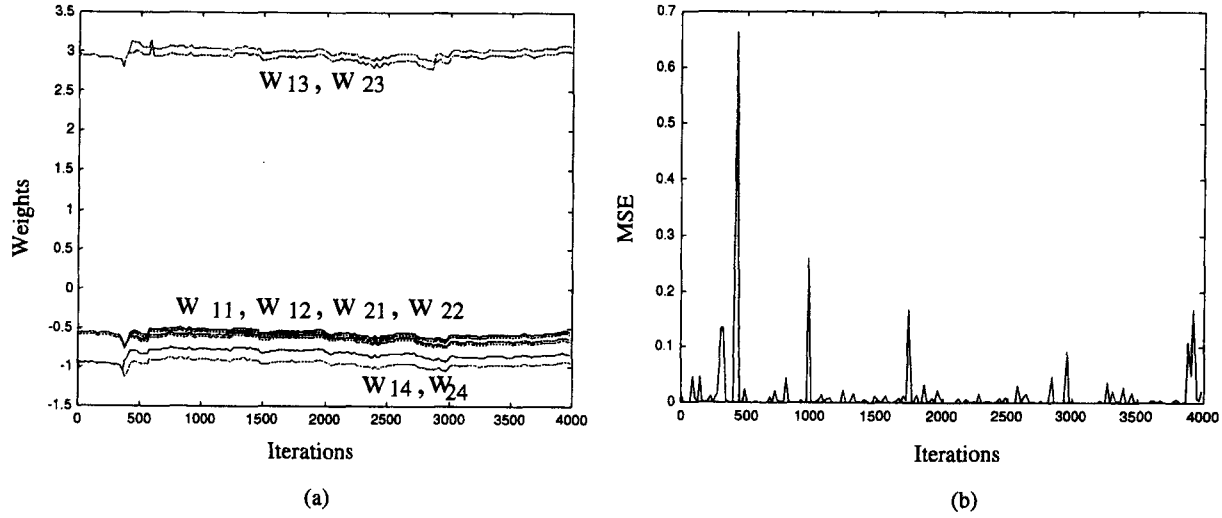


Fig. 18. Variation curves of anti-burst-interference algorithm 2 with burst-interference: (a) Weight values, (b) MSE.

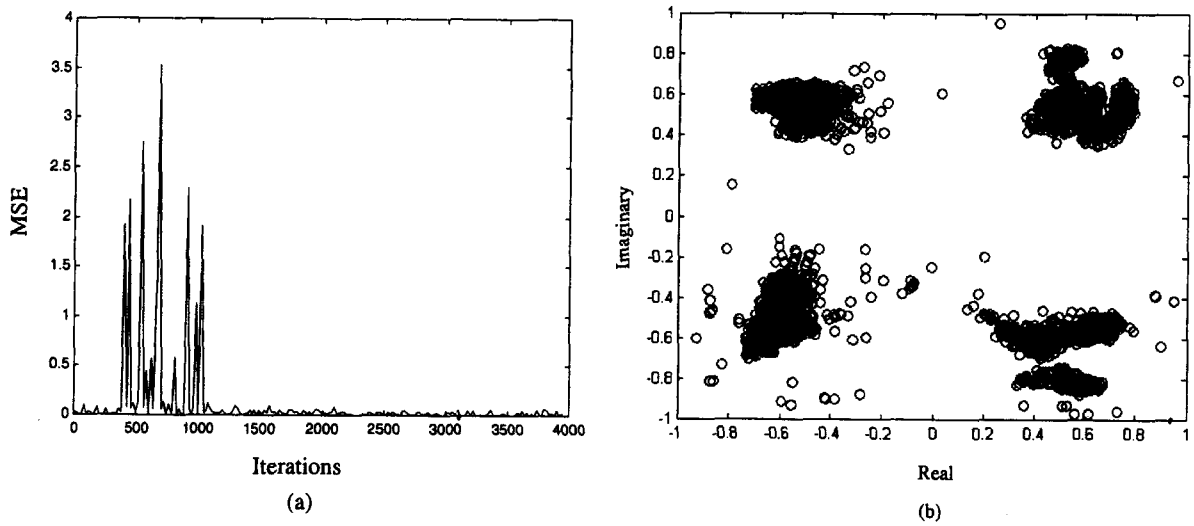


Fig. 19. Plots without anti-burst-interference measure: (a) MSE, (b) signal constellation.

distortion channels, but also exerts the predominance of the decision feedback structures that can refrain from ISI. It possesses better equalization performance than CRNNE.

Moreover, the anti-burst-interference algorithms can effectively prevent false convergence when channel meets burst-interference, and make equalizer automatically recover normal working conditions and continue to work. Thereby, the algorithms amplify the compatibility of system. Furthermore, it is fit for both real and complex case. We are currently investigating the modified algorithms to be used with other neural network equalizers.

REFERENCES

[1] G. J. Gibson, S. Siu, and C. F. N. Cowan, "Application of multilayer perceptrons as adaptive channel equalizers," in *Proc. IEEE Internat. Conf. Acoust. Speech Signal Process*, Glasgow, Scotland, May 1989, pp. 1183–1186.

[2] I. Cha and S. A. Kassam, "Channel equalization using adaptive complex radial basis function network," *IEEE J. SAC*, vol. 13, no. 1, pp. 121–131, Jan. 1995.

[3] M. J. Bradley and P. Mars, "Application of recurrent neural networks to communication channel equalization," *IEEE Trans. Neural Networks*, vol. 5, no. 2, Mar. 1994.

[4] G. Kechriotis, E. Zervas, and E. Manolakos, "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Trans. Neural Networks*, vol. 5, no. 2, pp. 617–620, Mar. 1994.

[5] G. Kechriotis and E. S. Manolakos, "Training fully recurrent neural networks with complex weights," *IEEE Trans. Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 41, no. 3, pp. 235–238, March 1994.

[6] H. Jiang and K. S. Kwak, "On neural networks adaptive equalizer for digital communication," *J. Korean Institute Commun. Sciences*, vol. 26, no. 10A, pp. 1639–1644, 2001.

[7] H. Ping *et al*, "Adaptive equalization technique application to time-variant fading channels," *ACTA ELECTRONICA SINICA*, vol. 21, no. 4, pp. 85–89, Apr. 1993.

[8] G. M. Georgiou and C. Koutsougeras, "Complex domain backpropagation," *IEEE Trans. Circuits and Syst.-II*, vol. 39, pp. 330–334, May 1992.

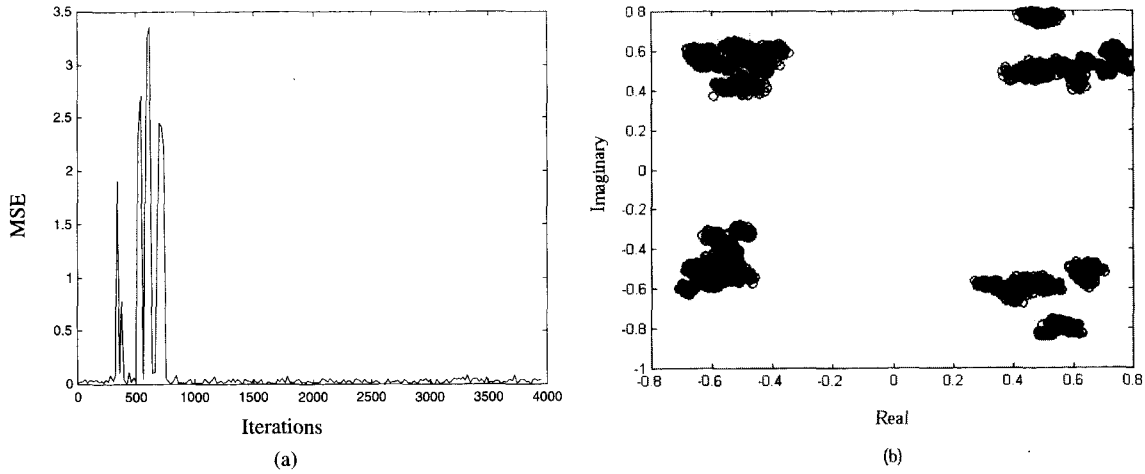


Fig. 20. Plots with anti-burst-interference algorithm 1: (a) MSE, (b) signal constellation.

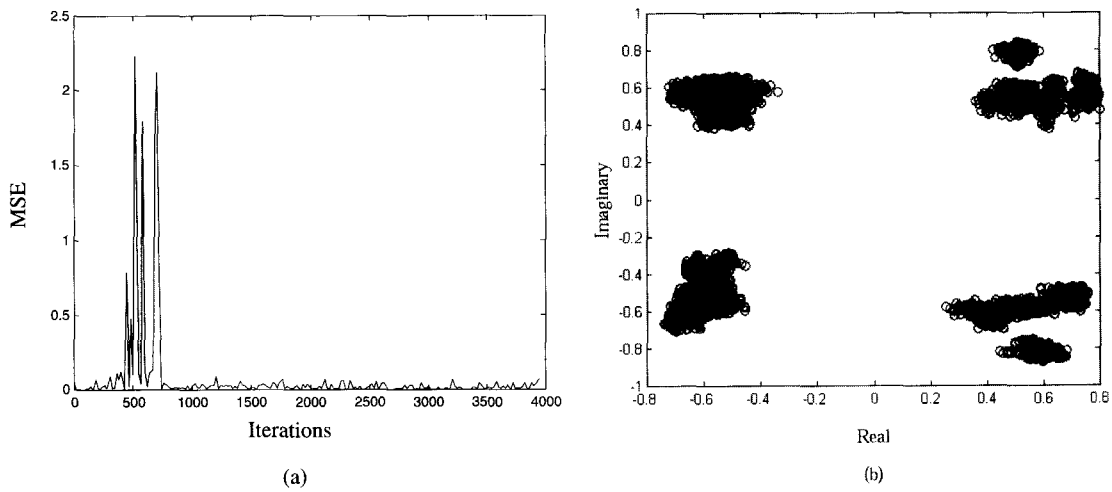
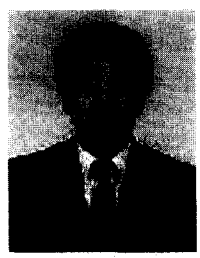


Fig. 21. Plots with anti-burst-interference algorithm 2: (a) MSE, (b) signal constellation.

[9] J. Cho, C. You, and D. Hong, "The neural decision feedback equalizer for the nonlinear digital magnetic recording systems," *ICC '96*, Dallas, Texas, U.S.A., vol. 1, pp. 573–576, 1996.

Hongrui Jiang received the BS degree in Microwave Telecommunication Engineering from Xidian University, Xian City, China in 1997, and the MS degree in Processing of Signal & Information from Guilin Institute of Electronic Technology, Guilin City, China in 2000, and the Ph.D degree in electronic telecommunication engineering from Inha University, Korea in 2003. His research interests include neural networks and telecommunication.



Kyung Sup Kwak received the B.S. degree from the Inha University, Inchon, Korea in 1977, and the M.S. degree from the University of Southern California in 1981 and the Ph.D. degree from the University of California at San Diego in 1988, under the Inha University Scholarship and the Korea Electric Association Scholarship Grants, respectively. From 1988 to 1989, he was a Member of Technical Staff at Hughes Network Systems, San Diego, California. From 1989 to 1990, he was with the IBM Network Analysis Center at Research Triangle Park, North Carolina. Since then, he has been with Inha University, Korea as a professor. He had been the associate dean of the School of Electrical and Computer Engineering from 1999 to 2000 and the dean of the Graduate School of Information Technology and Telecommunications from 2001 to 2002 at the Inha University, Inchon, Korea. Since 1994, he had been serving as a member of Board of Directors, and he is now the executive vice president for Korean Institute of Communication Sciences(KICS). In 1993, he received Engineering College Young Investigator Achievement Award from Inha University and a distinguished service medal from the Institute of Electronics Engineers of Korea (IEEK). In 1996 and 1999, he received distinguished service medals from the KICS. He received the Inha University Engineering College Paper Award and the LG/KICS Paper Award in 1998, and Motorola Paper Award in 2000. His research interests include multiple access communication systems, mobile and satellite communication systems, data networks, wireless Internet. He is members of IEEE, IEICE, KICS and KIEE.