Using Recurrent Neural Networks for Adaptive Communication Channel Equalization

G. Kechriotis, E. Zervas, and E. S. Manolakos, *Member, IEEE*

***Abstract*—Recently, nonlinear adaptive filters based on a variety of neural network models have been used successfully for system identification and noise-cancellation in a wide class of applications. An important problem in data communications is that of channel equalization, i.e., the removal of interferences introduced by linear or nonlinear message corrupting mechanisms, so that the originally transmitted symbols can be recovered correctly at the receiver. In this paper we introduce an adaptive Recurrent Neural Network (RNN) based equalizer whose small size and high performance makes it suitable for high-speed channel equalization. We propose RNN based structures for both trained adaptation and blind equalization, and we evaluate their performance via extensive simulations for a variety of signal modulations and communication channel models. It is shown that the RNN equalizers have comparable performance with traditional linear filter based equalizers when the channel interferences are relatively mild, and that they outperform them by several orders of magnitude when either the channel's transfer function has spectral nulls or severe nonlinear distortion is present. In addition, the small-size RNN equalizers, being essentially generalized IIR filters, are shown to outperform multilayer perceptron equalizers of larger computational complexity in linear and non-linear channel equalization cases.**

I. Introduction

THE DEMAND for very high speed efficient data transmission over physical communication channels has been substantially increased during the last decade. Communication channels are usually modeled as linear filters having a low-pass frequency response. If the amplitude and the envelope delay response are not constant within the bandwidth (non-ideal filters), the channel distorts the transmitted signal in both amplitude and delay, causing what is known as *intersymbol interference* (ISI). As a result of this linear distortion the transmitted symbols are spread and overlapped over successive time intervals. In addition to linear distortion the transmitted symbols are subject to other impairments such as thermal noise, impulse noise and *non-linear* distortion arising from the modulation/demodulation process, crosstalk interference, the use of amplifiers and converters, and the nature of the channel itself. All the signal processing techniques used at the receiver's end to combat the introduced channel distortion and recover the transmitted symbols are referred to as *equalization* schemes.

Adaptive equalization is characterized in general by the structure of the equalizer, the adaptation algorithm and the use or not of training sequences [1]. Linear equalization employs a linear filter usually with a FIR or lattice structure. A Recursive Least Squares (RLS) algorithm or a stochastic gradient algorithm, such as the Least Mean Squares (LMS), is used to optimize a performance index. When the channel has a deep spectral null in its bandwidth, linear equalization performs poorly since the equalizer places a high-gain at the frequency of the null, thus enhancing the additive noise at this frequency band. Decision Feedback Equalization (DFE) [1] can be employed to overcome this limitation. Although DFE and other methods, such as the Maximum Likelihood (ML) sequence detection [2], are nonlinear, the nonlinearity lies in the way the transmitted sequence is recovered at the receiver with the channel model being linear. If nonlinear channel distortion is too severe to ignore the aforementioned algorithms suffer from a severe performance degradation.

Among the techniques that have been proposed to address the non-linear channel equalization problem are those in [3], [4], [5], that rely on the Volterra series expansion of the nonlinear channel. In [6], [7] the authors used a feedforward neural network, a highly nonlinear structure, for the equalization of linear and nonlinear channels. This neural network equalizer was trained to approximate the correct mapping from delayed channel outputs to originally transmitted symbols, and it was shown that significant performance improvements can be achieved.

In this paper we propose the use of a *Recurrent Neural Network* (RNN) equalizer for the adaptive equalization of linear and nonlinear channels. RNNs have feedback, a property which makes them attractive for the equalization of nonlinear channels with deep spectral nulls. It is shown that RNNs of reasonable size have the ability to accurately model the inverse of a communication channel with a performance superior than that of the traditional equalization algorithms. A novel training approach is introduced for *blind equalization* of nonlinear channels, using only a partial set of statistics of the transmitted signal. Blind equalization is a particularly useful and difficult type of equalization when training sequences are undesirable or not feasible, as for example in the case of multipoint communication networks. In the absence of a training sequence, the only knowledge about the transmitted signal is the constellation from which the symbols are drawn. Blind equalization schemes such as the Sato [8], the Godard [9], the Tricepstrum Equalization Algorithm (TEA) [10], the Maximum Likelihood joint data and channel estimation [11], [12], algorithms that exploit the cyclostationarity property of the transmitted signal, have been developed for linear

Manuscript received February 2, 1993; revised June 28, 1993.

The authors are with the Communications and Digital Signal Processing (CDSP) Center for Research and Graduate Studies, Electrical and Computer Engineering Department, Northeastern University, Boston, MA 02115.

IEEE Log Number 9214802.

channels. The use of these algorithms with nonlinear unknown channels is questionable. Recurrent Neural Networks on the other hand, with their ability to learn nonlinear mappings of arbitrary complexity, may be proved invaluable towards the solution of the challenging non-linear channel blind equalization problem.

The rest of this paper is organized as follows: In Section II, the problem of communication channel equalization is formulated. In the same section we briefly describe two traditional algorithms, the *Recursive Least Squares* (Kalman, RLS) [1] and the *Constant Modulus Algorithm* (CMA) [13], for trained adaptation and blind equalization respectively, which we will use for comparisons with the proposed equalization schemes. In Section III we briefly review the basic principles of Recurrent Neural Networks as well as the *Real-Time Recurrent Learning* (RTRL) algorithm [14] and the Complex Real-Time Recurrent Learning (CRTRL) [15] we used for their training. In Section IV we describe the proposed trained adaptation RNN based equalizer and we evaluate its performance via simulations for both linear and nonlinear channels. Sections V and VI discuss the RNN blind equalizer and present simulation results for several linear and nonlinear channels as well. Finally, Section VII summarizes our findings and ends by pointing to further research directions.

II. Problem Statement: Existing Algorithms

The problem of channel equalization can be formulated as follows: A sequence of symbols $\boldsymbol{x} = \{x[0], x[1], x[2], \cdots\}$ is transmitted through a channel $\boldsymbol{h}$. The channel $\boldsymbol{h}$ is modeled either as a linear operator, in which case the output of the channel $\boldsymbol{y}$ is simply the convolution of the input sequence $\boldsymbol{x}$ with $\boldsymbol{h}$, ($\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x}$), or as a nonlinear operator, in which case we denote the output of the channel as $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})$. The channel noise is usually modeled as additive zero mean Gaussian, such that the input sequence at the receiver is $\hat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{n}$. The purpose of the equalizer $\boldsymbol{c}$ is to reconstruct the originally transmitted sequence $\boldsymbol{x}$, or at least a delayed and/or phase shifted version of it. Thus, in the absence of noise, ideal equalization implies that $\hat{\boldsymbol{x}} = \boldsymbol{c}(\hat{\boldsymbol{y}}) = \boldsymbol{\delta}_D * (e^{j\theta}\boldsymbol{x})$, where θ is a constant phase shift, and $\delta_D[t] = 1$ for $t = D$ and $\delta_D[t] = 0$ for $t \neq D$.

Existing equalization techniques employ a linear filter equalizer $\boldsymbol{c}$,whose coefficients are being adjusted to match the channel characteristics. Depending on whether the equalizer knows the originally transmitted sequence $\boldsymbol{x}$ or not, it is characterized as *trained adaptation* or *blind equalizer* respectively. The most widely used algorithm for linear trained adaptation equalizers is the Recursive Least Squares (RLS), or Kalman algorithm. A linear filter of sufficient length N is used at the output of the channel, such that the estimate of the transmitted symbol at time t is given by:

$$\hat{x}[t] = \sum_{k=0}^{N-1} c_k \hat{y}[t-k] = \boldsymbol{c}^T \hat{\boldsymbol{y}}_e[t]$$

where, $\boldsymbol{c} = [c_0 \; c_1 \; \cdots \; c_{N-1}]^T$ and $\hat{\boldsymbol{y}}_e[t] = [\hat{y}[t] \; \hat{y}[t-1] \; \cdots \; \hat{y}[t-N+1]]^T$. During the adaptation period, at every time instance t, the equalizer's error $e[t] = x[t] - \hat{x}[t]$ is calculated along with the Kalman gain vector $\boldsymbol{k}[t]$ and the inverse of the correlation matrix $\boldsymbol{P}[t]$, via the recursive equations:

$$\boldsymbol{k}[t] = \frac{\boldsymbol{P}[t-1]\hat{\boldsymbol{y}}_e^*[t]}{w + \hat{\boldsymbol{y}}_e^T[t]\boldsymbol{P}[t-1]\hat{\boldsymbol{y}}_e^*[t]} \tag{1}$$

$$\boldsymbol{P}[t] = \frac{1}{w}\left[\boldsymbol{P}[t-1] - \boldsymbol{k}[t]\hat{\boldsymbol{y}}_e^T[t]\boldsymbol{P}[t-1]\right] \tag{2}$$

where $0 < w < 1$ is the *forgetting factor*, "$*$" denotes here the complex conjugate and "T" denotes the transpose of a vector. Finally the equalizer's coefficients are updated via:

$$\boldsymbol{c}[t] = \boldsymbol{c}[t-1] + \boldsymbol{k}[t]e[t] \tag{3}$$

The RLS algorithm exhibits faster convergence than the Least Mean Squares (LMS) algorithm, at the expense of greater computational complexity.

In blind equalization, the equalizer has no exact knowledge of the transmitted sequence. The adaptation of the equalizer is attempted in a way to match the *statistics of the output* of the equalizer to those of the transmitted sequence. Existing algorithms for blind equalization, are based on the result due to Benveniste *et al.* [16] stating that if the channel is linear and the equalizer is an infinite length FIR filter, ideal equalization is achieved if the distribution of the transmitted sequence $\boldsymbol{x}$ is identical to that of the output of the equalizer $\hat{\boldsymbol{x}}$, provided that $x[t]$ is not Gaussian. If the input signal is white and the channel is linear and minimum phase, linear prediction can be used at the receiver to decorrelate the received data and establish the whiteness of the transmitted signal. However, when the linear channel $\boldsymbol{h}$ is non-minimum phase, linear prediction will match only the amplitude of the transfer function but not its phase. To extract the phase characteristics of the channel from the received data, it is necessary to use higher-order statistics of the received signal.

The majority of the proposed blind equalization algorithms employ a nonlinear function of the output of the equalizer to give rise to those higher order statistics. Among them is the *Constant Modulus Algorithm* (CMA) which adjusts the linear equalizer taps $\boldsymbol{c}$ in order to minimize the cost function:

$$J_p(\boldsymbol{c}) = \frac{1}{2p}E\{(|\hat{x}|^p - R_p)^2\}, \; p = 1, 2, \cdots \tag{4}$$

The algorithm penalizes deviations of $|\hat{x}|^p$ from a constant modulus R_p that depends on the input constellation $\left(R_p = \frac{E\{|x[t]|^{2p}\}}{E\{|x[t]|^p\}}\right)$. The case of $p = 2$ corresponds to the well known Godard algorithm [9], in which the equalizer's coefficients are updated using a stochastic gradient approach:

$$\boldsymbol{c}[t+1] = \boldsymbol{c}[t] - \gamma \cdot (|\hat{x}[t]|^2 - R_2) \cdot \hat{x}[t] \cdot \hat{\boldsymbol{y}}_e[t] \tag{5}$$

where $\hat{\boldsymbol{y}}_e[t]$ is the received data vector, defined in the same way as for the RLS case, and γ is a small positive constant that governs the rate of convergence. All the algorithms that are based on the minimization of one of the functions in (4), are guaranteed to achieve perfect equalization only if the global minimum of the objective function is reached and only under the assumption that the length of the equalizer is infinite.

However, all the objective functions in (4) are non-convex and therefore possess local minima where the algorithm might get trapped [17]. Moreover, although these algorithms perform well for channels with relatively flat frequency response, their performance is much worse when the linear channel has zeros close to the unit circle (spectral nulls).

If the nonlinearities in the channel are too severe to ignore, the above mentioned algorithms fail to sufficiently equalize the channel since the equalizer is modeled as a linear filter. In such cases a nonlinear equalizer is preferable over a linear one. However, if a nonlinear equalizer is used, minimizing one of the family of objective functions (4), might not guarantee perfect equalization. It can be shown [18] that under certain conditions, if the combined channel-equalizer system admits a Volterra series representation and the transmitted symbols are binary, the minimization of a modified objective function guarantees the perfect reconstruction of the transmitted sequence.

III. Recurrent Neural Networks

Recurrent Neural Networks (RNNs), in which every unit is connected to every other unit, are the most general case of neural networks. RNNs are highly nonlinear dynamical systems that exhibit a rich and complex dynamical behavior. They have been proven better than traditional signal processing methods in modeling and predicting nonlinear and chaotic time series [19] and in a wide variety of applications ranging from speech processing and adaptive channel equalization [20], [21], to modeling finite state automata and learning simple grammar rules [22]. In [23] it has been shown that there is a direct correspondence between the Volterra series representation of a nonlinear dynamical system and a RNN, and it has been proven that given any dynamical system there exist a finite size RNN whose first and second order Volterra kernels are equal to those of the original system.

A RNN has in general m external inputs and n fully interconnected units. A RNN with $m = 1$ and $n = 3$ is shown in Fig. 1. The activation of any (or all) of the units in the network can be considered as the output of the network and all the units can be trained to produce desired outputs.

A RNN is a *dynamical* system. The output of a unit at time $t+1$ depends not only on the external inputs to the network $x_l^{net}[t], l = 1, \cdots, m$ at the previous time instant, but also on the previous outputs of the units $y_l[t]$, $l = 1, \cdots, n$. The dynamics of the RNN are described by the following set of equations:

$$s_k[t+1] = \sum_{l=1}^{n} w_{kl}[t] y_l[t] + \sum_{l=1}^{m} w_{k,l+n}[t] x_l^{net}[t] \quad (6)$$

$$y_k[t+1] = f(s_k[t+1]) \quad (7)$$

where $w_{ij}[t]$ is the the weight of the connection from the j^{th} to the i^{th} unit at time t, and the *activation* function $f(\cdot)$ can be any real function differentiable with respect to its argument. (Usually $f(\cdot)$ is taken to be the hyperbolic tangent function $\tanh(\cdot)$).

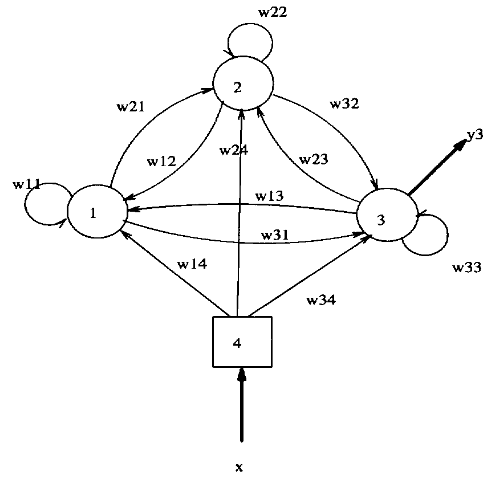


Fig. 1. A Recurrent Neural Network (RNN) with $n = 3$ units (oval nodes), and $m = 1$ input nodes (square nodes). The x is the external input of the RNN and y_3 the output.

From the equations describing the dynamics, we see that a RNN models a nonlinear IIR filter, since every output depends on *all* the previous outputs of the network. In contrast, a feedforward neural network whose inputs come from taped delay lines, can only model a nonlinear FIR filter with memory at most equal to the number of the delayed neural network inputs.

Several algorithms have been proposed for the training of the RNNs. The most widely known algorithm is the Real-Time Recurrent Learning (RTRL) algorithm, proposed by Williams and Zipser [14], that can be used to update the weights of the RNN in real time. The RTRL algorithm is summarized bellow:

1) Forward Phase: For $k = 1, \cdots n$

Compute the output of the k^{th} neuronat time $t+1$ using (6) and (7).

2) Learning Phase: If $d_k[t+1]$, $k = 1, \cdots n$, is the desired output of the k^{th} unit at time $t+1$, the error at the k^{th} unit is given by:

$$e_k[t+1] = d_k[t+1] - y_k[t+1]$$

It has been assumed without loss of generality that there exist desired values for all units in the RNN. The *instantaneous* total error at time $t+1$ is given by:

$$J[t+1] = \frac{1}{2} \sum_{k=1}^{n} e_k^2[t+1] \quad (8)$$

and the objective is to change the weights in the direction that minimizes $J[t+1]$. If we define, as in [14], the "sensitivity:"

$$p_{ij}^k[t] = \frac{\partial y_k[t]}{\partial w_{ij}}$$

then the training phase involves the following steps:

1) Evaluate $p_{ij}^k[t+1]$, $i = 1, \cdots, n$ $j = 1, \cdots, n+m$, $k = 1, \cdots, n$ recursively using:

$$p_{ij}^k[t+1] = f'(s_k[t+1]) (\sum_{l=1}^{n} w_{kl}[t] p_{ij}^l[t] + \delta_{ik} x_j[t]) \quad (9)$$

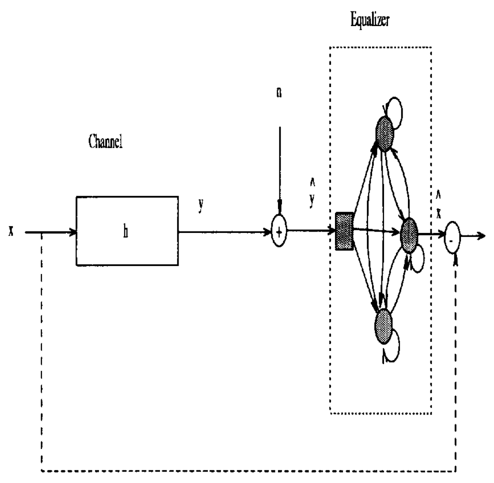


Fig. 2. The block diagram of the communication system with RNN equalizer. If the RNN is used as a trained adaptation equalizer, the dashed line is active. If the RNN is used as a blind equalizer, the dashed line is removed.

where δ_{ik} is the Kronecker delta and $f'(\cdot)$ denotes the derivative of $f(\cdot)$.

2) Update the weights in the direction of the steepest descent:

$$w_{ij}[t+1] = w_{ij}[t] + \alpha \sum_{k=1}^{n} e_k[t+1] p_{ij}^k[t+1] \qquad (10)$$

where $\alpha > 0$ is the *learning rate* (i.e., the step of the gradient descent algorithm).

In many cases in communication systems, the inputs and outputs of a channel are best described as complex valued signals and the channel is modeled as a linear or nonlinear complex operator. In [15], the RTRL algorithm has been extended for the case of a RNN whose inputs, outputs, weights and activation functions are complex. The complex RTRL algorithm (CRTRL) is summarized in the Appendix.

IV. The RNN Trained Adaptation Equalizer

The block diagram of a communication system that employs a RNN trained adaptation equalizer is shown in Fig. 2. The transmitter sends a *known training sequence to the receiver*, and the receiver adjust itself so that it reproduces the correct transmitted symbols. Then, the adaptation stops and the transmitter sends the data. During the training or adaptation period, at every time instant t, the error between the output of the RNN equalizer $\hat{x}[t]$ and the originally transmitted symbol $x[t]$ is formed, and the weights of the RNN are adjusted via the RTRL algorithm. In the case of PAM signals [1] and communication channels with real coefficients, the RTRL algorithm is used to compute the gradient of the squared error with respect to the weights of the equalizer, and in the case of either QAM signals [1] or channels with complex coefficients the CRTRL algorithm is used. As shown in Fig. 2, the input to the RNN equalizer at time t, $\hat{y}[t]$, is equal to the channel output $y[t]$ plus Gaussian noise $n[t]$. Note here that an RNN with more than one input can also be used, to model more complex channels.

The weights and activations of the RNNs we used as trained adaptation channel equalizers, were initialized to small random values. In all cases, the RNN equalizers had a small number of units (usually two or three). This is in contrast with the much larger number of units of the feedforward neural network based channel equalizers that have been proposed before (In [6] the authors used a 3-layer perceptron with structure [2 9 3 1] to equalize linear and nonlinear communication channels).

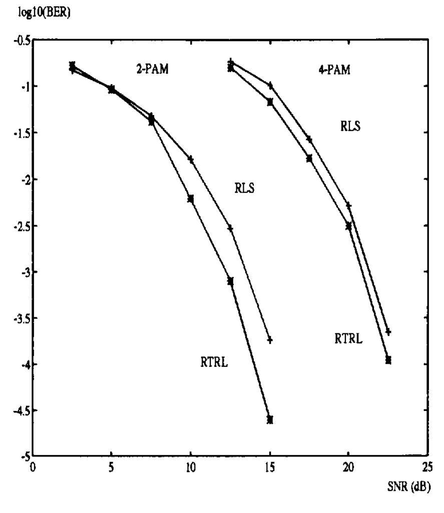
A. Linear Channels

We first tested the performance of the proposed RNN-based trained adaptation equalizer for a simple linear minimum phase channel with a relatively flat frequency response (no zeros near the unit circle). For such channels, linear equalizers whose coefficients are being adapted via the RLS algorithm are known to perform very well. The transfer function of the channel was: $H_1(z) = 1 + 0.7z^{-1}$ and a one-input, two units, one-output RNN was compared against a linear equalizer of length 20.

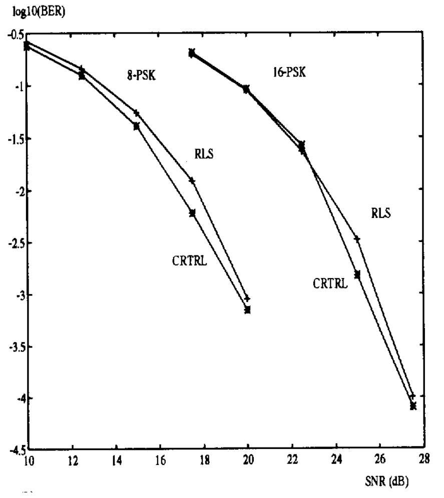
The Bit-Error-Rate (BER),defined as the ratio of misclassified to correct symbols at the output of the equalizer, was evaluated for 2-PAM, 4-PAM, 8-PSK and 16-PSK signals. The RTRL algorithm was used to train the RNN equalizer for the 2-PAM and 4-PAM case, whereas for the PSK signals we used complex RNNs trained via the CRTRL. We simulated 100 different realizations for each value of the SNR. For each realization, the weights of the RNN equalizer were initialized to small random values satisfying $|w_{ij}| < 10^{-3}$ and so where the initial activations of the units. The value of the learning rate was kept constant at $\alpha = 0.5$ and the pseudo-random input and noise sequences were generated with different seeds for the random number generators. The RNN and RLS based equalizers were first trained with 2000 symbols from the output of the channel and then we evaluated the BER based on 10^4 more received symbols, for each realization.

In Fig. 3 we plot the decimal logarithm of the BER achieved by the linear and the nonlinear equalizers. As we can see, even for that simple linear channel, for which the linear equalizers exhibit a very good performance, the nonlinear RNN equalizers achieve comparable or even smaller BER. The largest performance improvement is achieved for the case of 2-PAM binary signals but even for higher constellations the RNN equalizer achieves lower bit error rates than the linear equalizer. This is not only due to the fact that the RNN equalizer, essentially being an IIR type of filter, can approximate the inverse of the FIR channel more accurately than the truncated FIR filter, but also due to the ability of the RNN equalizer to form nonlinear decision regions. In noisy environments the optimal decision regions are described by nonlinear functions [6], and thus the better performance of the RNN can be explained.

The convergence speed of the RNN equalizer depends on both the channel characteristics and the value of the learning rate α(adaptation step). In Fig. 4 we show the normalized MSE at the output of the RNN and RLS equalizers averaged over 100 realizations for the case of a 2-PAM signal with SNR equal to 15 dB. The MSE curves of the RNN equalizer are plotted for several values of the learning rate. As we can see



(a)



(b)

Fig. 3. Linear channel $H_1(z) = 1 + 0.7z^{-1}$. Plot of the decimal logarithm of the BER achieved by a linear filter equalizer and an RNN-based equalizer versus the value of the SNR. 100 realizations per SNR value were used.

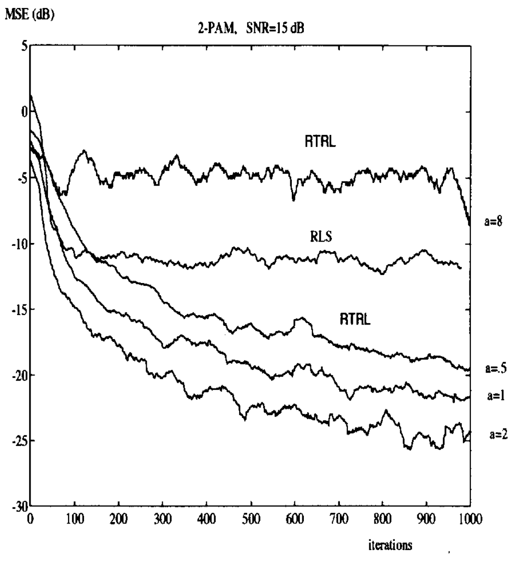


Fig. 4. Linear channel $H_1(z) = 1 + 0.7z^{-1}$. Plot of the normalized MSE at the output of the linear and RNN-based equalizers (for different a) vs. the number of iterations.

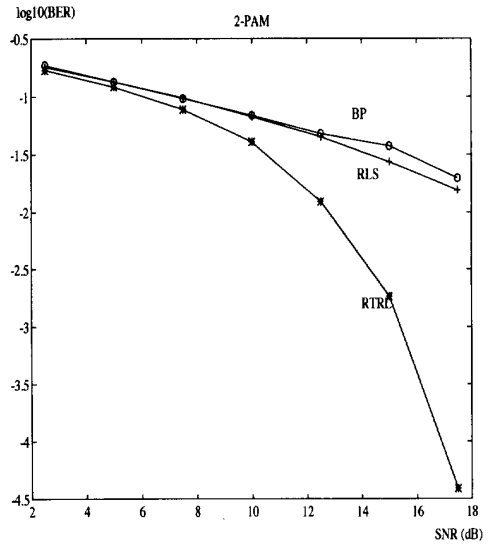
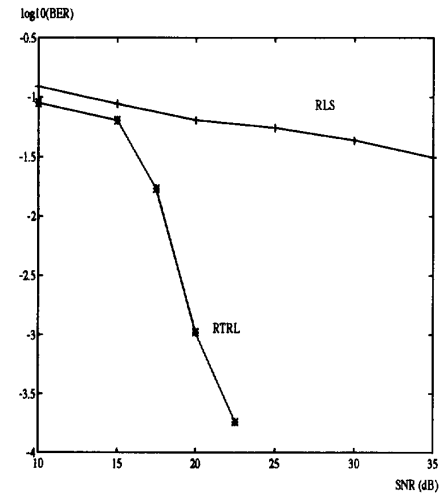


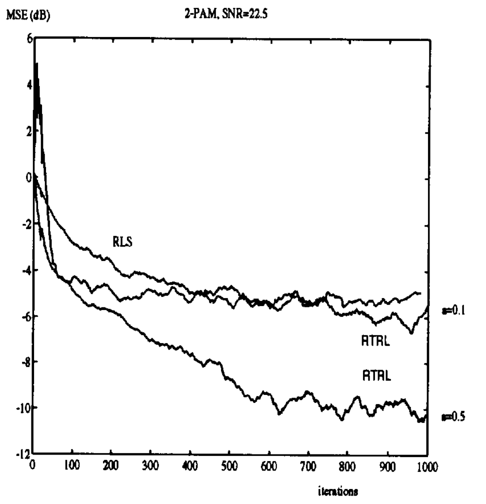
Fig. 5. Linear channel $H_2(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$. 2-PAM. BER comparison of linear, feedforward neural network and RNN equalizers. 100 realizations per SNR value were used.

from Fig. 4, the RTRL exhibits the same very fast convergence as the RLS algorithm, provided that the learning rate is chosen appropriately. Small values of the learning rate result in slower convergence, whereas large values might cause instability (see case with $\alpha = 8$).

In Fig. 5 we show the bit-error-rates achieved by the linear and RNN based equalizers for a non-minimum phase channel whose transfer function is given by: $H_2(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$. Such channels are more likely to be encountered in a practical communication system [6], [7]. The RNN and linear equalizers have the same structure as for the previous example and the transmitted signal was a 2-PAM sequence. As we can see from Fig. 5, the two-unit RNN equalizer achieves bit-error-rates as much as three-orders of magnitude smaller than the ones that the linear equalizer achieves. For the same linear channel we also investigated the performance of a 5-9-3-1 feedforward neural network (FNN) [7] trained to learn the mapping from delayed outputs of the channel to the correct transmitted symbol. The performance of the FNN, whose weights have been adapted via the back propagation (BP) algorithm with learning rate equal to $\alpha = 1$ and momentum term $\beta = 0.7$, has been found to be similar to that of the performance of the linear equalizer for the range of SNRs shown. All three equalizers were trained to follow the transmitted sequence delayed by one sample.



(a)



(b)

Fig. 6. Partial response linear channel $H_3(z) = 1 - 2z^{-1} + z^{-2}$, 2-PAM. (a) BER comparison of the linear and RNN based equalizers, 100 realizations per SNR value were used. (b) The normalized MSE at the output of the linear and RNNequalizer (for $a = 0.1$ and $a = 0.5$).

B. Partial Response Channels

The transfer function of partial response channels has zeros on the unit circle. Such channel are frequently encountered in magnetic recording [24] and since the inverse of the channel is not defined, there exists no linear filter that would sufficiently equalize them. Therefore nonlinear methods have to be used to reconstruct the originally transmitted signal.

The performance of the proposed RNN based equalizer was investigated for the case of a 2-PAM signaling scheme, for a partial response channel whose transfer function was: $H_3(z) = 1 - 2z^{-1} + z^{-2}$. This channel has a double zero on the unit circle and therefore, as expected, the linear (20-taps) transversal filter equalizer exhibits very poor performance. As shown in Fig. 6, where we plot the BER achieved by the linear equalizer and by a two-unit RNN whose initial settings were the same as for the previous cases, the nonlinear equalizer outperforms the linear one by as much as three orders of magnitude for the range of SNR shown. In the same figure, we also show the MSE achieved by each of the equalizers versus the number of the iterations for a value of the SNR fixed at 22.5 dB. Again, it is evident that the convergence speed of the RNN equalizer is comparable to that of the RLS algorithm if the value of the learning rate is chosen appropriately. The inability to determine a priori the optimal learning rate is a significant problem, not only for the RNN case but for the linear equalizers as well (LMS algorithm [1]).

Simulations with larger RNNs show a performance comparable to that ofthe two-unit RNN. It might be conjectured that since a two-unit, one input, one output RNN is essentially a six degree of freedom ($n^2 + m$ weights) nonlinear IIR model it is sufficient to model any FIR channel with six or less coefficients. In most practical situations it may be assumed that the intersymbol interference spans predominantly at most in the order of six symbols so that the two-unit RNN is sufficient to model many communication channels encountered in practice.

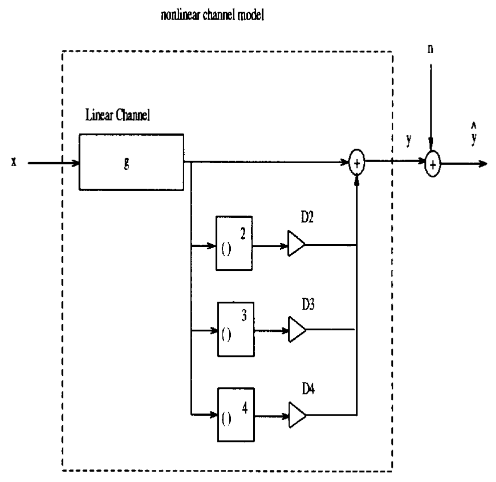


Fig. 7. The model of the nonlinear communication channel used.

C. Nonlinear Channels

The nonlinear channel used in our simulations, has the structure of the model shown in Fig. 7: The transmitted sequence is passed through a linear channel whose transfer function is $G(z)$, and the output of that channel is added to nonlinear harmonics. The value of the gain coefficients D_2, D_3, and D_4 determine how severe the nonlinear distortion will be. We first simulated a channel whose linear sub-channel component has the transfer function: $G(z) = 1+0.7z^{-1}$. Such nonlinear channel models are frequently encountered in data transmission over digital satellite links, especially when the signal amplifiers operate in their high-gain limits.

The performance of the RNN-based equalizer was compared to the performance of a linear 20-taps transversal equalizer trained with the RLS algorithm. Again, a two-unit RNN was used as the trained adaptation equalizer and the initial settings

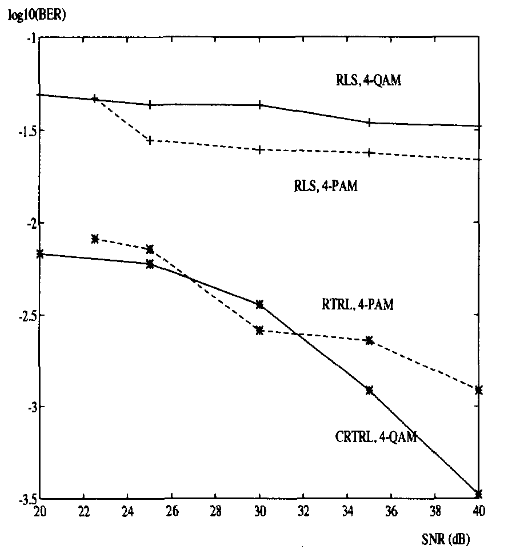


Fig. 8. Nonlinear Channel Equalization, trained adaptation. BER comparison of the linear and the RNN equalizer when $G(z) = 1 + 0.7z^{-1}$, and the additive nonlinearity gains are: For 2-PAM, $D_2 = 1, D_3 = 0.7, D_4 = 0.5$ and for 4-QAM, $D_2 = 0.6, D_3 = 0.5, D_4 = 0.4$.

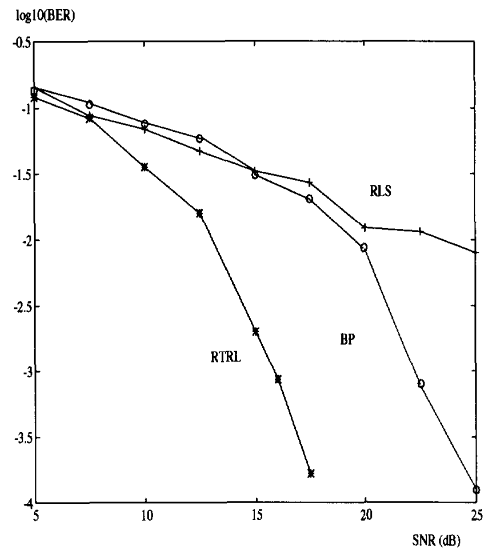


Fig. 9. Nonlinear Channel Equalization, trained adaptation. BER comparison of the linear, FNN, and RNN equalizers when $G(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$, 2-PAM signaling and $D_2 = 0.2$, $D_3 = D_4 = 0$.

were the same as before. We evaluated the achievable BER by both equalizers for the 4-PAM and the 4-QAM case. For the 4-PAM case the additive nonlinearity gains were: $D_2 = 1.0$, $D_3 = 0.7$, $D_4 = 0.5$ and both the inputs to the network and the training symbols have been scaled such that their range is in the interval [-1,+1]. In the 4-QAM case the coefficients have been set to: $D_2 = 0.6$, $D_3 = 0.5$, $D_4 = 0.4$. As we can see from Fig. 8, in which we plot the BER versus the value of the SNR, the RNN equalizer clearly outperforms the linear equalizer by as much as two orders of magnitude. This was expected since the linear equalizer can compensate only for the linear channel distortion.

In another example, the linear subchannel has the transfer function $G(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$, the input was 2-PAM and $D_2 = 0.2$, $D_3 = D_4 = 0$. We compared the performance of a linear equalizer of length 20, a feedforward neural network of structure 5-9-3-1 and a two-unit RNN equalizer. From Fig. 9, where we plot the BER versus the value of the SNR, we see that both the RNN and the FNN equalizers outperform the linear equalizer but the BER drops faster with increasing SNRs for the case of the RNN.

In Fig. 10, we plot the "eye patterns" (output values) of both the linear and RNN equalizer applied to the same channel, for SNR=30dB and 4-QAM signaling. Both equalizers have been trained first using 1000 symbols. Then 2000 more symbols were transmitted and the receiver's output was plotted. As we can see from Fig. 10, the linear equalizer completely fails to open the eye-pattern, whereas the symbols are clearly distinguishable at the output of the RNN equalizer.

V. The Recurrent Neural Network Blind Equalizer

A RNN, being essentially an IIR nonlinear filter, can be trained to have desired dynamical behavior, using a stochastic gradient approach, via the RTRL algorithm. If the output of the equalizer (RNN in our case) is exactly the same as the transmitted signal (with a possible time delay, and/or phase shift) then it should have the same moments as the transmitted signal.

Both in the linear and in the nonlinear channel case, the combination of the channel and the RNN equalizer results in a nonlinear system whose inputs are the transmitted symbols $x[t]$ and its outputs are the estimates $\hat{x}[t]$. As it has been shown in [23] such a nonlinear system admits a triangular kernel Volterra series expansion form:

$$\hat{x}[t] = \sum_{l=0}^{\infty} \sum\sum \cdots \sum_{1\le k_1 \le k_2 \le \cdots \le k_l}^{\infty} h_l(k_1, k_2, \cdots, k_l) \cdot x[t-k_1]x[t-k_2]\cdots x[t-k_l] \tag{11}$$

If the transmitted symbols are binary assuming the values $\{-1,\ 1\}$ and are drawn from an i.i.d zero mean distribution, then it can be shown [18] that the Volterra series expansion form (11) may be simplified to:

$$\hat{x}[t] = \sum_{l=0}^{\infty} \sum\sum \cdots \sum_{1< k_1 < k_2 < \cdots < k_l}^{\infty} h_l(k_1, k_2, \cdots, k_l) \cdot \cdot x[t-k_1]x[t-k_2]\cdots x[t-k_l] \tag{12}$$

i.e.,. for every term $h_l(k_1, k_2, \cdots, k_l)$, no two indices among $k_1, k_2, \cdots, k_l$ are the same. Furthermore if $E\{\hat{x}[k]\} = 0$, $E\{\hat{x}^2[k]\} = 1$, $E\{\hat{x}^4[k]\} = 1$, then all but one of the coefficients $h_l(k_1, k_2, \cdots, k_k)$ in (12) have to be equal to zero [18]. If, as usually the case for real-life communication channels, the magnitude of the coefficients of orders two and higher is smaller than that of the first order coefficients, then the estimated sequence of symbols $\hat{x}[k]$ is just a delayed (and possibly rotated by a phase θ) version of the originally transmitted sequence $x[k]$.

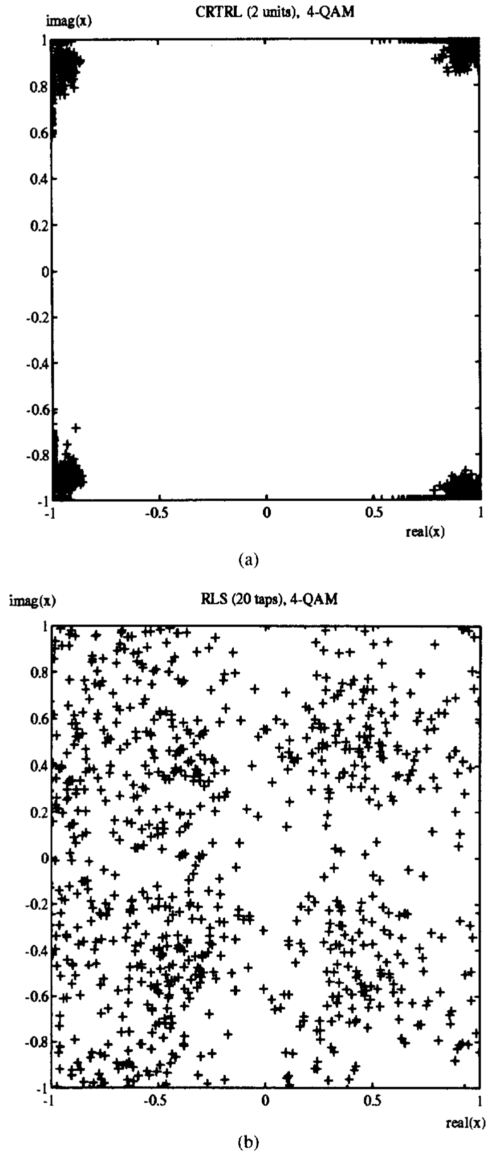


Fig. 10. Eye-patterns of the linear and RNN equalizers, 2000 iterations after the end of training. The same non-linear channel of the previous figure was used.

We would like to minimize, at time $t+1$, the objective function:

$$\mathcal{E}[t+1]=\sum_{k=1}^{4}\alpha_k e_k[t+1] \;=\; \sum_{k=1}^{4}\alpha_k(E_{t+1}\{\hat{x}^k\}-E\{x^k\})^2 \tag{13}$$

where, E_{t+1} denotes the estimated mean value using the $t+1$ first outputs of the RNN, and α_k are positive constants that define the *weight* of the corresponding term e_k in the objective function (13). We would like to derive a way to update the weights of the RNN, depending on the output at time $t+1$, namely $\hat{x}[t+1]$, so that the objective function (13) gets minimized.

Each of the four mean values in (13) can be computed recursively using averaging as:

$$E_{t+1}\{\hat{x}^k\}=\frac{1}{t+1}(tE_t\{\hat{x}^k\}+\hat{x}^k[t+1]) \tag{14}$$

Differentiating $\mathcal{E}[t+1]$ with respect to the current weights w_{ij} we obtain:

$$\frac{\partial\mathcal{E}}{\partial w_{ij}}[t+1]=$$
$$\frac{2}{t+1}\left\{\sum_{k=1}^{4}\alpha_k(E_{t+1}\{\hat{x}^k\}-E\{x^k\})k\hat{x}^{k-1}[t+1]\right\}\frac{\partial\hat{x}[t+1]}{\partial w_{ij}} \tag{15}$$

Now $\frac{\partial\hat{x}[t+1]}{\partial w_{ij}}=p_{ij}^{1}[t+1]$ can be computed recursively using (9). Therefore, the algorithm for the minimization of the objective function (13) with a Recurrent Neural Network via the RTRL becomes:

1) Initialize the estimates for $E_0\{\hat{x}^k\}$ to zero for $k=1,2,3,4$.
2) Present a new sample of the channel output to the RNN input. Compute the RNN $\hat{x}[t+1]$ using (6) and (7).
3) Update the moment estimates $E_{t+1}\{\hat{x}^k\}$, $k=1,\ 2,\ 3,\ 4$, using (14).
4) Compute the gradient with respect to the weights of the previous time step, using (9) and (15).
5) Update the weights in the direction of the steepest descent with learning rate α.
6) Go to Step 2, unless the objective function has been sufficiently minimized.

VI. RNN Blind Equalization Simulations

In our computer simulations we evaluated the performance of the proposed RNN blind equalizer for several linear and nonlinear communication channels and the results were compared to those obtained with a linear CMA equalizer based on the Godard criterion (equation (4), for $p=2$). As in the case of trained adaptation equalization, the RNNs we used were fully interconnected with a small number of units (two or three), one input and one output. In all simulations the weights and activations of the RNN were initialized to small random values. The RTRL algorithm was employed to train the weights of the RNN. The performance measures used for the comparison of the linear transversal equalizer and and the proposed nonlinear RNN-based equalizer were the eye-patterns and the normalized MSE at the output of the equalizers, as well as the BER for a range of SNRs. The MSE and the BER have been averaged over 100 independent realizations for each case. For each realization, the weights and the activations of the RNN have been initialized randomly and different pseudo-random input and noise sequences have been employed.

The value of the adaptation step (learning rate) for the RTRL algorithm was kept fixed to $\alpha=2.5$ for all realizations, whereas the parameter γ in the adaptation equation (5) for the linear equalizer was set to $\gamma=0.01$. The length of the linear equalizer was 31. In all simulations the input sequence $x[k]$ is a 2-PAM signal. To calculate the BER, the RNN was first trained with an 1000 symbols long sequence. Then the adaptation of the weights was stopped and an 10 000 long

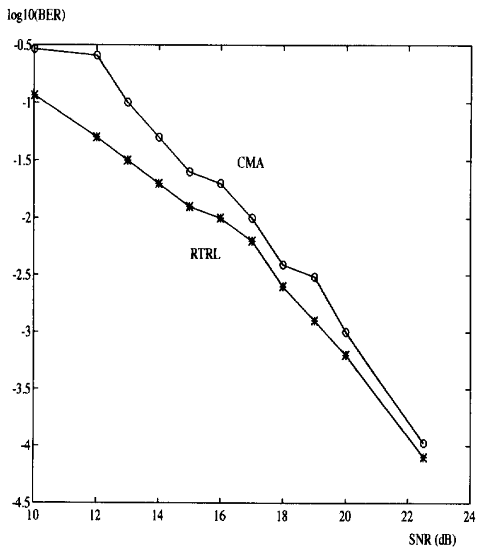


Fig. 11. BER vs. SNR for the RNN and CMA blind equalizers. Linear mixed-phase channel $H_3(z) = (1-2.0z^{-1})(1-0.6z^{-1})$, 2-PAM signaling.

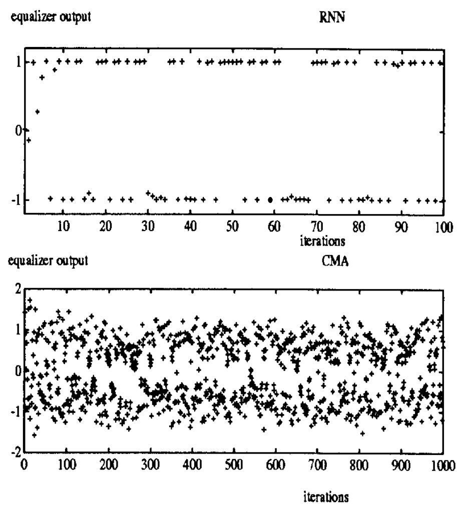
sequence of received symbols was fed into the RNN to test its performance. For the linear equalizer case, the training was done with 10 000 symbols and then the BER calculation was based on 10 000 more received symbols.

A. Mixed-Phase Channels

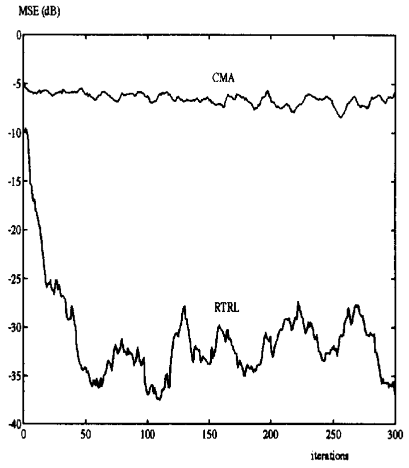
The linear mixed phase channel with transfer function $H_3(z) = (1 - 2.0z^{-1})(1 - 0.6z^{-1})$ was used first to evaluate the performance of the proposed RNN blind equalizer. The RNN equalizer has two units, one input and one output. The weights of the RNN were initialized to random values satisfying $|w_{ij}| < 0.01$, and the learning rate was $\alpha = 2.5$. The values of the coefficients α_k in the objective function (13) were: $\alpha_1 = 2$, $\alpha_2 = 10$ $\alpha_3 = 0$, and $\alpha_4 = 10$.

In Fig. 11 we compare the performance of the RNN equalizer and the linear one by plotting the BER versus the value of the SNR. As we can see, even for this linear channel model, the nonlinear equalizer performs better than the linear one, especially at low SNR's due to its ability to form decision regions closer to the optimal than its linear counterpart. For higher values of the SNR the difference in the relative performance of the two equalizers becomes smaller, and eventually a linear equalizer might be preferable because of its reduced complexity. In the high noise region, both equalizers are more likely to be trapped in a local minimum. In our experiments we observed surprisingly that the frequency of encountering a local minimum was higher for the CMA equalizer than for the RNN. Intuitively, this might be due to the small size of the RNN that was used (for the one input, two-unit RNN objective function, the minimization has to carried out with respect to only six parameters, versus 31 parameters for the linear equalizer).

In Fig. 12 we plot the eye-pattern at the output of both the linear and the nonlinear equalizer and the averaged, over 100



(a)



(b)

Fig. 12. (a) The output of the RNN and CMA blind equalizers for the linear mixed-phase channel $H_3(z) = (1 - 2.0z^{-1})(1 - 0.6z^{-1})$, SNR = 20 dB, vs. the number of iterations. (b) MSE (in dB) averaged over 100 realizations, vs. the number of iterations.

realizations, normalized MSE curves for SNR = 20 dB. After about only 20 iterations, the RNN nonlinear equalizer opens the eye-pattern completely, whereas the CMA equalizer needs to be trained with at least 200 symbols to open the eye-pattern. The fast convergence of the RNN blind equalizer makes it suitable for use in environments where small initial adaptation delays are desired (such as mobile communications, cellular telephony).

The normalized MSE curves are plotted in Fig. 12(a). As we can see the MSE of the RNN equalizer is almost three orders of magnitude smaller than that of the CMA equalizer. This is because, in the 2-PAM case, the RNN equalizer models not only the inverse of the linear channel but also the thresholding device at the output.

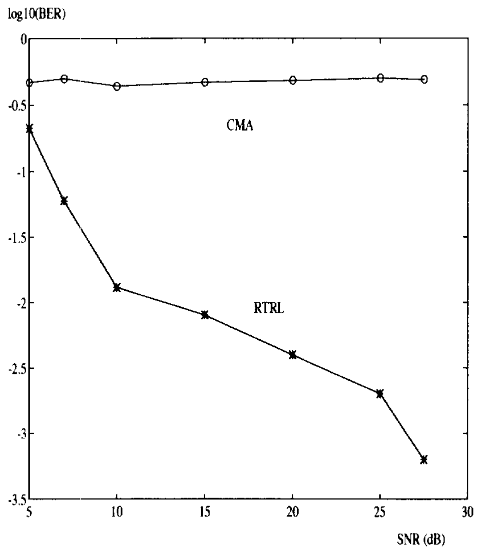


Fig. 13. BER comparison of the RNN and the CMA blind equalizers for nonlinear channel. The linear subcomponent is $G(z) = 1 + 0.7z^{-1}$, and the additive nonlinearity gains $D_2 = 0.15$, $D_3 = 0.10$, $D_4 = 0.05$, SNR = 30 dB.

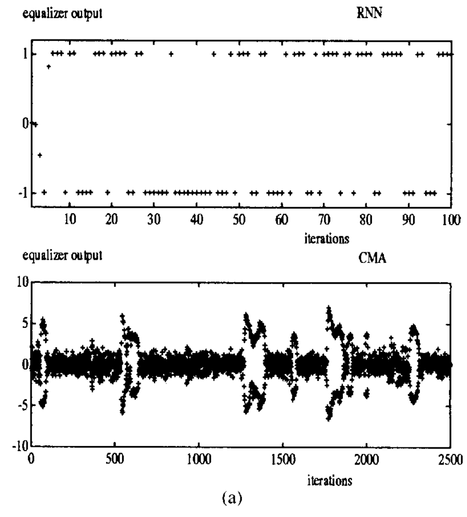
B. Nonlinear Channels

The model of the nonlinear channel used for the simulations, is the same as for the trained adaptation case. The linear component of the channel has transfer function $G(z) = 1 + 0.7z^{-1}$, and the values of the coefficients D_k are: $D_2 = 0.15$, $D_3 = 0.10$, $D_4 = 0.05$. A one input, one output, two units RNN was trained to minimize objective function (13), with $\alpha_1 = 2$, $\alpha_2 = 4$, $\alpha_3 = 4$, and $\alpha_4 = 4$. The learning rate for the weights adaptation was $\alpha = 2.5$.

Even for this relatively simple communication channel, the CMA based equalizer, being designed to compensate only for linear channel distortion exhibits a very poor performance. As we can see from Fig. 13 in which we plot the BER curves for both equalizers, the CMA equalizer exhibits an almost constant high error rate for the range of SNR shown. The two-unit RNN equalizer, as for the linear channel case, succeeds to open the eye-pattern in less than 15 iterations as shown in Fig. 14(b) where we plot the output values versus the number of iterations for 30 dB SNR. In Fig. 14(b) the averaged MSE of both equalizers is plotted versus the number of iterations. The RNN blind equalizer, outperforms the linear CMA equalizer by almost five orders of magnitude.

VII. Conclusions—Future Research Directions

We have shown that nonlinear adaptive filters based on RecurrentNeural Networks can be used for both trained adaptation and blind equalization of linear and nonlinear communication channels. Since RNNs essentially model nonlinear infinite memory filters, they can accurately realize, with a relatively small number of parameters, the inverse of finite memory systems, and thus compensate effectively for the channel introduced interferences. Extensive simulation results show that small size RNNs outperform the linear transversal filter equalizers in several cases.



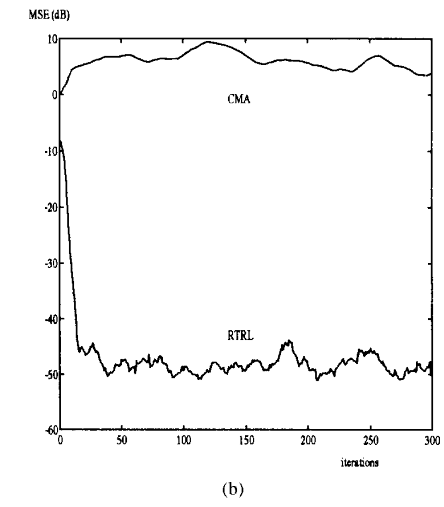


Fig. 14. (a) Output of the RNN and CMA blind equalizers. The channel was nonlinear with linear subcomponent $G(z) = 1 + 0.7z^{-1}$, and additive nonlinearity gains $D_2 = 0.15$, $D_3 = 0.10$, $D_4 = 0.05$, SNR = 30 dB. (b) The MSE (in dB) averaged over 100 realizations for each equalizer.

Although the computational complexity of the algorithms used to adjust the weights of RNNs is large (in the order of $O(n^4)$ for the RTRL algorithm, where n is the number of units), their small size makes them attractive for high-speed channel equalization in contrast to other proposed nonlinear equalizers including the feed-forward neural networks that need a much larger number of units to model the inverse of the channel. As an example, a digital implementation of the feedforward neural network equalizer of structure 5-9-3-1 that was proposed in [7] would have a computational complexity of 922 floating point operations (FLOPS) per incoming symbol during the training phase, whereas the complexity of a one-input, two-unit RNN equalizer is only 144 FLOPS per symbol (It has been assumed that the evaluation of the sigmoid function requires the equivalent of 10 floating point multiplications or additions time). It is interesting to note that the computational complexity of such small RNNs is even comparable to the fastest variants of the RLS algorithm (205 FLOPS per symbol are required for a linear equalizer of length 10 trained with the Fast RLS algorithm [1]) Moreover, trainable RNNs can be implemented in digital VLSI hardware in a highly parallelized fashion (see for example [25]), and even faster analog implementations are envisioned such that the supported data rates can be further increased.

RNNs can be used as blind equalizers when their objective function is suitably modified to penalize deviations of the statistics of their output from the statistics of the originally transmitted sequence. Although their performance seems to be much better than conventional blind equalization techniques for many different classes of channels, it is widely known that the training of RNNs is a nonlinear nonconvex optimization problem and there is always a chance that the training stochastic gradient descent algorithms can get trapped into local minima. Suitable initialization and search techniques can be possibly employed to assist convergence to the global minimum.

In a practical situation, it is possible to use the proposed nonlinear RNN-based equalizer in conjunction with a conventional linear equalizer. When the channel has a relatively flat spectrum and its linear nature dominates, the linear equalizer is slower than the RNN equalizer but less likely to be trapped in a local minimum. However, when the nonlinear distortion of the channel is too severe to ignore, or the channel has spectral nulls then the linear equalizer fails and the RNN equalizer performs remarkably better.

Recently, new methods for blind equalization of linear channels based on higher-order spectra have been developed [10], but although their objective function is convex they fail when the channels are predominantly nonlinear. The incorporation of higher order output statistics or cross moments in the objective function of the proposed RNN blind equalizers is currently under investigation. Another point that might further improve the convergence properties and facilitate the escape from local minima is to allow the coefficients α_k in the objective function (13) to be adjusted dynamically, and investigate systematic ways for their adaptation.

VIII. Appendix A

The Complex Real-Time Recurrent Learning Algorithm (CRTRL) is summarized below:

1) Forward Phase: For $k = 1 \cdots n$

$$S_k[t+1] = \sum_{l=1}^{n} W_{kl} Y_l[t] + \sum_{l=n+1}^{n+m} W_{kl} X_l[t] \tag{16}$$

$$Y_k[t+1] = F(S_k[t+1]) = f(s_{Rk}[t+1]) + jf(s_{Ik}[t+1]) \tag{17}$$

where now all inputs, outputs, and weights are complex numbers and $f(\cdot)$ is again the $tanh(\cdot)$ function as for the real case. In (17), the subscripts R and I denote the real and imaginary parts of a complex number correspondingly and $j \overset{\text{def}}{=} \sqrt{-1}$.

2) Retrieving Phase: Defining the sensitivitiesP_{RR}, P_{RI}, P_{IR}, P_{II} as:

$$\begin{bmatrix} P^k_{RRij} & P^k_{RIij} \\ P^k_{IRij} & P^k_{IIij} \end{bmatrix}[t] \overset{\text{def}}{=} \begin{bmatrix} \frac{\partial y_{Rk}}{\partial w_{Rij}} & \frac{\partial y_{Rk}}{\partial w_{Iij}} \\ \frac{\partial y_{Ik}}{\partial w_{Rij}} & \frac{\partial y_{Ik}}{\partial w_{Iij}} \end{bmatrix}[t] \tag{18}$$

the recursive equations for the computation of the sensitivity terms become:

$$\begin{bmatrix} P^k_{RRij} & P^k_{RIij} \\ P^k_{IRij} & P^k_{IIij} \end{bmatrix}[t+1] = \begin{bmatrix} f'(s_{Rk}) & 0 \\ 0 & f'(s_{Ik}) \end{bmatrix}[\text{t+1}] \times$$

$$\times \left(\sum_{l=1}^{n} \begin{bmatrix} w_{Rkl} & -w_{Ikl} \\ w_{Ikl} & w_{Rkl} \end{bmatrix} \begin{bmatrix} P^l_{RRij} & P^l_{RIij} \\ P^l_{IRij} & P^l_{IIij} \end{bmatrix}[\text{t}] + \right.$$

$$\left. + \delta_{ik} \begin{bmatrix} x_{Rj} & -x_{Ij} \\ x_{Ij} & x_{Rj} \end{bmatrix}[t] \right) \tag{19}$$

Finally, the weight update equation becomes:

$$W_{ij} = W_{ij} + \alpha \ \nabla_{W_{ij}} J[t] = W_{ij} + \tag{20}$$

$$+\alpha \sum_{k=1}^{n} \left(\begin{bmatrix} e_{Rk} & e_{Ik} \end{bmatrix} \begin{bmatrix} P^k_{RRij} & P^k_{RIij} \\ P^k_{IRij} & P^k_{IIij} \end{bmatrix} \begin{bmatrix} 1 \\ j \end{bmatrix} [\text{t}] \right)$$

References

[1] J. Proakis, *Digital Communications*. Englewod Cliffs, NJ: Prentice Hall Inc., 1988.

[2] G. D. Forney, Jr., "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," in*Proc. IEEE, Trans. Infor. Theory*, vol. 18, pp. 378–383, 1972.

[3] S. Benedetto and E. Biglieri, "Nonlinear equalization of digital satellite channels," *IEEE Journal on Sel. Areas in Comm.*, vol. 1, pp. 57–62, 1983.

[4] E. Biglieri, A. Gersho, R. D. Gitlin, and T. L. Lim, "Adaptive cancellation of nonlinear intersymbol iinterference for voiceband data transmission," *IEEE Journal on Sel. Areas in Comm.*, vol. 2, pp. 765–777, 1984.

[5] D. D. Falconer, "Adaptive equalization of channel nonlinearities in QAM data ttransmittion systems," *Bell System Technical Journal*, vol. 57, no. 7, pp. 2589–2611, 1978.

[6] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant, "Adaptive equalization of finite non-linear channels using multilayer perceptrons," *Signal Processing*, vol. 20, pp. 107–119, 1990.

[7] G. J. Gibson, S. Siu, and C. F. N. Cowan, "Multilayer perceptrons structures applied to adaptive equalisers for data communications," in *ICASSP Int. Conf. Accoustics, Speech and Signal Proc.*, Glasgow, Scotland, May 1989, pp. 1183–1186.

[8] Y. Sato. "A method of self-recovering equalization for multilevel amplitude modulation," *IEEE Trans. Commun.*, vol. 23, pp. 679–682, 1975.

[9] D. N. Godard, "Self-recovering equalization and carrier tracking in two dimensional data communication systems," *IEEE Trans. Commun.*, vol. 28, pp. 1867–1875, 1980.

[10] D. Hatzinakos and C. L. Nikias, "Blind equalization using a tricepstrum based algorithm," *IEEE Trans. Commun.*, 1991.

[11] N. Seshadri. "Joint data and channel estimation using fast blind trellis search techniques," in *Proc. Globecom*, pages 1659-1663, 1990.

[12] E. Zervas, J. G. Proakis, and V. Eyuboglu, "Blind equalization approaches for shaped transmitted signals." *Submitted to IEEE Trans. on Communications*, October 1992.

[13] J. R. Treichler and B. G. Agee, "A new approach to multipath correction of constant modulus signals," *IEEE Trans. on Accoustics, Speech and Signal Proc.*, vol. 31, no. 2, 1983.

[14] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, pp. 270–280, 1989.

[15] G. Kechriotis and E. S. Manolakos, "Training fully recurrent neural networks with complex weights," *IEEE Transactions on Circuits and Systems-II*, to appear.

[16] A. Benveniste, M. Goursat, and G. Ruget, "Robust identification of a nonminimum phase system: Blind adjustment of a linear equalizer in data communications." *IEEE Trans. Automat. Control*, vol. 25, pp. 385–399, 1980.

[17] Z. Ding and R. A. Kennedy, "On the whereabouts of local minima for blind adaptive equalizers," *IEEE Trans. Circuits and Systems*, vol. 39, pp. 119–123, 1992.

[18] G. Kechriotis, "On the blind equalization of nonlinear communication channels," in preparation, 1993.

[19] G. Kechriotis and E. Manolakos, "Using neural networks for nonlinear and chaotic signal processing," in *ICASSP Int. Conf. Accoustics, Speech and Signal Proc.*, April 1993, pp. 465–469.

[20] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," Technical Report TR-I-0006, Advanced Telecommunications Research Institute, Japan, 1987.

[21] G. Kechriotis, E. Zervas, and E. Manolakos. "Using recurrent neural networks for blind equalization of linear and nonlinear communication channels," in *IEEE Conference on Military Communications*, San Diego, CA, October 1992, pp. 784–788.

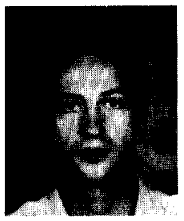
[22] D. Servan-Schreiber, A. Cleermans, and J. L. McClelland, "Encoding sequential structure in simple recurrent networks," Carnegie-Mellon Internal Report, CMU-CS-88-183, 1988.

[23] N. Z. Hakim, J. J. Kaufman, G. Gerf, and H. E. Meadows, "Volterra characterization of neural networks," In *Proc. IEEE Asilomar Conference on Signals, Systems and Computers*, November 1991, pp. 1128–1132.

[24] H. Kobayashi and D. T. Tang, "Application of partial response channel coding to magnetic recording systems," *IBM Journal*, 368–375, July 1970.

[25] G. Kechriotis and E. S. Manolakos, "A VLSI array architecture for the on-line training of recurrent neural networks," in *Proc. IEEE Asilomar Conference on Signals, Systems and Computers*, November 1991, pp. 506–510.

George Kechriotis was born in Graz, Austria in 1965. He received the Diploma in electrical and computer engineering from the National Technical University of Athens, Greece in 1988 and a M.Sc. degree in electrical engineering from California Institute of Technology in 1989. Since 1990 he has been with Northeastern University, where he is currently a Ph.D. candidate in the electrical and computer engineering Deptartment. He is the author or coauthor of more than 10 refereed publications. His research interests include neural networks, parallel processing. and digital communications.



Evangelos Zervas received the Diploma in electrical engineering from the National Technical University of Athens, Greece. Then he joined the ECE Dept. of Northeastern University where he obtained the M.Sc. degree in 1989 and the Ph.D. degree in 1993, both in electrical and computer engineering. During his graduate studies at Northeastern he has been a research assistant with the Communications and Digital Signal Processing (CDSP) Center for Research and Graduate studies. Dr. Zervas is a member of the IEEE Communications and Signal Processing Societies. His research interests include digital communications, information theory, adaptive filtering, and neural networks. He is currently with the Greek Air Force Computing Center in Athens, Greece.

Elias S. Manolakos, (S'81–M'81–S'82–S'85–M'85–M'89), received the Ph.D. degree in Computer Engineering from the University of Southern California in 1989, the M.S.E.E. degree from University of Michigan, Ann Arbor and the Diploma in Electrical Engineering from the National Technical University of Athens, Greece. Since 1989 he has been an Assistant Professor with the Electrical and Computer Engineering Dept. of Northeastern University where he is leading the Parallel Processing and Architectures research group of the Communications and Digital Signal Processing (CDSP) Center for Research and Graduate studies. Dr. Manolakos was the recipient of Greek State Scholarships Foundation award in Computer Science and Engineering in 1984, and National Science Foundation's Research Initiation Award in 1993. His research interests include parallel signal/image processing architectures, neural networks and fault tolerant computing.