

# A RNN-LC Hybrid Equalizer

*Magno T. Madeira da Silva and Max Gerken*

Department of Telecommunications and Control Engineering

Escola Politécnica - University of São Paulo - Brazil

e-mail: {magno, mgk}@lcs.poli.usp.br

## ABSTRACT

A hybrid equalizer using a linear combiner and a recurrent neural network is presented. It characterizes itself for being adaptive and presenting: 1) a worst case performance very close to the best of the substructures that composes it, being better than each one of them in critical situations; 2) a computational complexity that makes its implementation feasible; and, 3) a good performance in difficult environments as, for example, channels with non-minimum phase, spectral nulls or non-linearities. Adaptation of coefficients is done using both the LMS and RTRL algorithms. Simulations illustrate the good performance of the proposed equalizer.

## 1 Introduction

Equalization of communication channels can be interpreted as a non-linear classification problem that demands adaptive solutions. This fact motivates the use of non-linear structures such as neural networks [1]. In many situations, they may present a better performance in comparison with conventional equalizers as Linear Transversal Equalizers (LTE) and Decision-Feedback Equalizers (DFE) which, having a lower computational complexity, present a sub-optimal performance.

Optimal equalizers based on the Bayes criterion or on the Maximum Likelihood criterion (for example the Viterbi algorithm) have high computational complexity, a limitation that is shared by equalizers based on neural networks like MLP (Multilayer Perceptrons), RBF (Radial Basis Function) or SOFM (Self-Organized Maps). Usually, these networks have high complexity and demand long training times.

Thus, a simple approach is to use less complex non-linear structures to improve the performance of conventional equalizers. Obviously, it's desirable to get structures which are feasible to implement.

With this in mind many authors have considered hybrid equalizers based on linear structures (LTE) and neural networks [1, 6]. Variations of these structures have also appeared using decision feedback. In [6] some hybrid structures were presented which basically consist of the LTE in series with a RBF or MLP network.

Although these equalizers present a good performance for non-linear channels, they have high computational complexities due to the use of RBF or MLP networks. In [4, 5] a Recurrent Neural Network (RNN) was proposed for equalization purposes. This network has low complexity and can be trained using the Real Time Recurrent Learning (RTRL) algorithm which updates the weights of the network in real time [5]. Comparing this equalizer with the LTE/DFE, it was observed in several situations that when one presents a poor performance, the other presents a good one. This behavior suggests combining these structures in order to obtain the best of both.

In this paper a hybrid equalizer formed by a DFE and a RNN is proposed. It is described in the next section. Thereafter, some simulation results are presented. A conclusion section closes the paper.

## 2 The Proposed RNN-LC Hybrid Equalizer

The proposed hybrid equalizer (Fig. 1) is formed by a RNN and a linear combiner (LC) connected with two delay lines constituting the DFE substructure. The input of this substructure is given by  $M_f$  delayed samples of the input signal  $u(n)$  and by  $M_b$  delayed samples of the RNN output. The inputs of the RNN substructure are given by the same samples of the input signal  $u(n)$  and by  $M_r$  past decisions that are fed back. In Fig. 1  $a(n)$  and  $\hat{a}(n)$  represent respectively the transmitted symbols and their estimates.

Figure 2 shows the RNN substructure with  $N = 3$  fully interconnected neurons and  $M$  external inputs. In this case the external inputs  $x_i(n)$ ,  $0 \leq i < M$ , are given by  $M_f$  delayed samples of the input signal  $u(n)$  and by  $M_r$  past decisions that are fed back. The output of a neuron at time  $n + 1$  depends not only on the external inputs  $x_i(n)$ ,  $0 \leq i < M$ , but also on the previous outputs of all the neurons ( $y_k(n)$ ,  $1 \leq k \leq N$ ). In Fig. 2 the existing delays at the connections between neurons and the sigmoidal non-linearity at each neuron output are omitted. More detailed descriptions of each one of these substructures can be found in [4] and [3].

On-line training of the considered hybrid equalizer can be performed using the RTRL algorithm [4], [5] for the RNN parameters and the LMS algorithm [2] for the LC weights. The problem that appears is how to obtain

---

This work was supported by FAPESP (proc. 00/12350-6) and CNPq (proc. 300521/92-8).

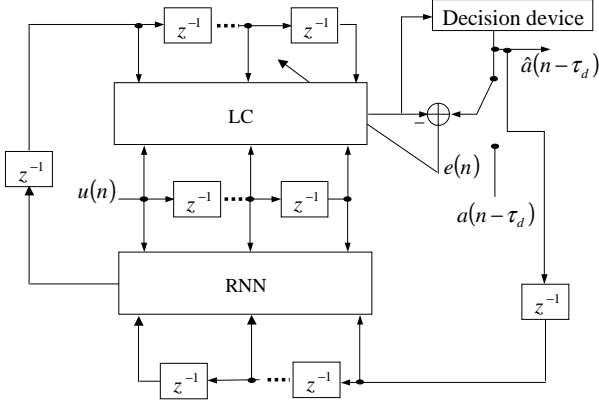


Figure 1: Scheme of RNN-LC hybrid equalizer.

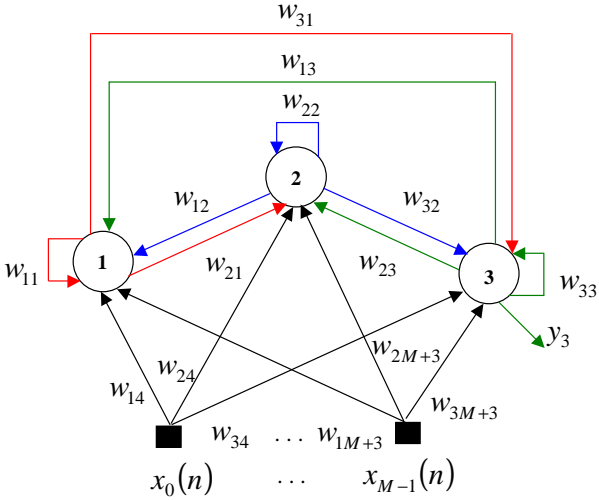


Figure 2: RNN with  $N = 3$ ,  $M$  inputs and output  $y_3$ .

the error at the RNN output, or in other words, how to propagate the error from the output of the LC to its inputs. Fortunately, due to the linear nature of the LC this problem has a very simple solution. Assuming that there is an error  $e_{\text{RNN}}(n)$  at the output of the RNN, Fig. 3 shows how this error propagates to the output of the LC. Thus, the error  $e(n)$  at the LC's output is given by

$$e(n) = \sum_{k=M_f}^{M_f+M_b-1} w_k(n) e_{\text{RNN}}(n - k + M_f),$$

where  $w_k$ ,  $k = M_f, M_f + 1, \dots, M_f + M_b - 1$ , are the LC weights associated to the input coming from the RNN,  $M_f$  is the number of inputs that constitute the channel output and  $M_b$  is the number of inputs that correspond to the RNN output. From the previous equation, the RNN error can be calculated as

$$e_{\text{RNN}}(n) = \frac{e(n) - \sum_{k=M_f+1}^{M_f+M_b-1} w_k(n) e_{\text{RNN}}(n - k + M_f)}{w_{M_f}(n)}.$$

From this expression one readily concludes that  $w_{M_f}(n) \neq 0$  must be satisfied. To guarantee this condition initially  $w_{M_f}(n) = 1$  was imposed. Moreover,

several simulations have shown that a further simplification, taking  $e_{\text{RNN}}(n) = e(n)$ , does not introduce significant performance deterioration as long as the extreme values of the sigmoidal non-linearities are compatible with the signals levels. Therefore, the computational

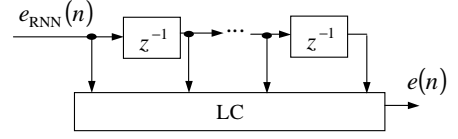


Figure 3: Error propagation for RNN adaptation.

complexity of the training algorithm is the addition of the computational complexities of the algorithms used for training each substructure under consideration: the RTRL algorithm for training the RNN and the LMS algorithm for training the LC. Table 1 shows the computational complexity of the RTRL and LMS algorithms for real signals. In this table, NL represents the non-linear operation of the sigmoidal functions. For complex signals appropriate versions of the RTRL [5] and the LMS [2] should be used.

To allow real time operation the training algorithm must present fast convergence. Simulations show that for most applications it is convenient to adjust the learning rates of both algorithms, RTRL and LMS, approximately equal, at least at the same order of magnitude. However, in some cases it was convenient to increase the learning rate of the RTRL to be about 10 times the learning rate of the LMS.

Op.	RTRL	LMS
$\times$	$(M_f + M_r)[N^2(N + 1) + 2] + (N + 1)(N^3 + 1) + 2N^2$	$2(M_f + M_b) + 1$
NL	$2N$	-
$+$	$(M_f + M_r)N(N^2 - N - 3) + (1 - N)(1 - N^3) + 3N^2$	$2(M_f + M_b)$

Table 1: Computational Complexity of the RTRL and LMS algorithms for real signals.

### 3 Simulations Results

The channel models used in the simulations are shown in Table 2. Coefficients  $h_0(n)$ ,  $h_1(n)$  and  $h_2(n)$  of channel  $\mathcal{H}_5$  are generated by passing a Gaussian white noise through a second order Butterworth filter designed to simulate a fade rate of 0.1 Hz [8]. The coefficients of the digital radio channels can be obtained from [7].

The performance of the hybrid equalizer has been investigated by measuring symbol error rates (SER) for 2-PAM and 4-QAM modulations. For the sake of comparison, DFE and RNN equalizers were also considered. The used configurations and the corresponding computational complexities of the training algorithms (for real signals) are shown in Table 3. Considering computational complexities of training algorithms, this table shows that the hybrid equalizer lies between the RNN equalizer with 2 and 3 neurons.

$\mathcal{H}_1$ (Digital radio channel) FIR model with 300 real coefficients from [7]
$\mathcal{H}_2$ ( $\mathcal{H}_1$ with non-linearities) $v_l(n) = \mathcal{H}_1(z)a(n)$ $v(n) = v_l(n) + 0.1v_l^2(n) + 0.05v_l^3(n)$
$\mathcal{H}_3$ (Time-variant [8]) $v(n) = a(n) + \text{sen}(2\pi n/T)a(n-1)$ , $T = 3000$
$\mathcal{H}_4$ ( $\mathcal{H}_3$ with non-linearities) $v_l(n) = a(n) + \text{sen}(2\pi n/T)a(n-1)$ $v(n) = v_l(n) + 0.2v_l^2(n) - 0.1v_l^3(n)$
$\mathcal{H}_5$ (Time-variant [8]) $v(n) = h_0(n)a(n) + h_1(n)a(n-1) + h_2(n)a(n-2)$
$\mathcal{H}_{1c}$ (Digital radio channel) FIR model with 300 complex coefficients from [7]
$\mathcal{H}_{2c}$ ( $\mathcal{H}_{1c}$ with non-linearities) $v_l(n) = \mathcal{H}_{1c}(z)a(n)$ $v(n) = v_l(n) + 0.05v_l^2(n) + 0.01v_l^3(n)$
Notation: $z^{-1}a(n) = a(n-1)$ $a(n)$ : channel input symbols $v(n)$ : noiseless channel output

Table 2: Communication channel models used in the simulations.

The considered delays of the training sequences were  $\tau_d = 50$  samples for the time-invariant channels ( $\mathcal{H}_1$ ,  $\mathcal{H}_2$ ,  $\mathcal{H}_{1c}$ ,  $\mathcal{H}_{2c}$ ), and  $\tau_d = 3$  samples for the time-variant channels ( $\mathcal{H}_3$ ,  $\mathcal{H}_4$ ,  $\mathcal{H}_5$ ).

Equalizer	$N$	$M_f$	$M_b$	$M_r$	$\times$	$+$	NL
RNN <sub>2</sub>	2	5	-	-	105	14	4
RNN <sub>3</sub>	3	5	-	-	320	94	6
DFE	-	5	4	-	19	18	0
HIB	2	5	4	3	166	28	4

Table 3: Used configurations and number of operations of the training algorithms for real signals.

Linear time-invariant channels have been used in figures 4 and 5. They show that in the worst case the hybrid equalizer presents a performance close to the best substructure (DFE) considered alone. In cases where the RNN equalizer has a better performance, the hybrid equalizer presents a performance close to it. Fig. 4-b and 5-b show that, in the case of non-linear channels, the hybrid equalizer outperforms the DFE and the RNN equalizer. Thus, its performance is not limited to the performance of their substructures.

In Fig. 6-a a linear time-variant channel is considered. The three equalizers present close performances for signal-to-noise ratios (SNR) below 15 dB. In this range the DFE performs slightly better than the hybrid equalizer, indicating that the RNN substructure has not properly converged. On the other hand, for SNR's above 15 dB, the hybrid equalizer presents the best performance. With the introduction of non-linearities (channel  $\mathcal{H}_4$ , Fig. 6-b) the hybrid equalizer outperforms the other two in the whole SNR range. Below SNR=15dB the behaviors are similar but above this value the hybrid equalizer shows its superiority.

Fig. 7 shows errors at the output of the decisors by considering the linear and time-variant channel  $\mathcal{H}_5$ . The considered signal-to-noise ratio is 20 dB. The absolute values of the roots of  $h_0(n)x^2 + h_1(n)x + h_2(n)$  are shown in Fig. 7-d so that burst of errors can be associated with rapid changes of these roots. Particularly, the bursts near iterations 5000 and 27000 are due to strong spectral nulls. In these situations, shown in Fig. 7, the hybrid equalizer presents a faster recuperation. Finally, it is worth noting that increasing the number of coefficients of the DFE does not result in significant improvements, at least for the examples under consideration. Therefore, a comparison with more complex DFEs has been omitted.

## 4 Conclusions

The proposed hybrid equalizer has reasonable computational complexity and presents good performance for difficult communication channels as, for example, channels with non-minimum phase, spectral nulls, non-linearities or a combination of these situations. By means of simulations it has been shown that the proposed equalizer outperforms Decision Feedback Equalizers and Equalizers based on Recurrent Neural Networks of similar computational complexities. More specifically, considering computational complexities of training algorithms, the hybrid equalizer lies between the RNN equalizers with 2 and 3 neurons.

## Acknowledgments

The authors thank Dr. Miguel A. Ramírez for his suggestions.

## References

- [1] M. Ibnkahla, "Applications of neural networks to digital communications - a survey," *Signal Processing*, Elsevier, vol. 80, pp. 1185-1215, 2000.
- [2] S. Haykin, *Adaptive Filter Theory*, third ed., New Jersey, Prentice Hall, 1996.
- [3] S. Haykin, *Neural Networks*, second ed., New Jersey, Prentice Hall, 1999.
- [4] G. Kechriotis, E. Zervas, and E. S. Manolakos, "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Trans. on Neural Networks*, vol. 5, pp. 267-278, Mar. 1994.
- [5] G. Kechriotis, and E. S. Manolakos, "Training fully recurrent neural networks with complex weights," *IEEE Trans. on CAS-II*, vol. 41, no. 3, pp. 235-238, March 1994.
- [6] S. Bouchired, D. Roviras, and F. Castani, "Equalisation of satellite mobile channels with neural network techniques," *Space Comm.*, vol. 15, pp. 209-220, 1998/1999.
- [7] The Signal Processing Information Base (SPIB). Signal Processing Society and NSF. Internet file (chan1.mat): <http://spib.rice.edu/spib/microwave.html> [Feb. 14, 2001].
- [8] T. Shimamura, and C. F. N. Cowan, "Equalisation of time variant multipath channels using amplitude banded techniques," in *Proc. IEEE ICASSP'1997*, pp. 2497-2500.

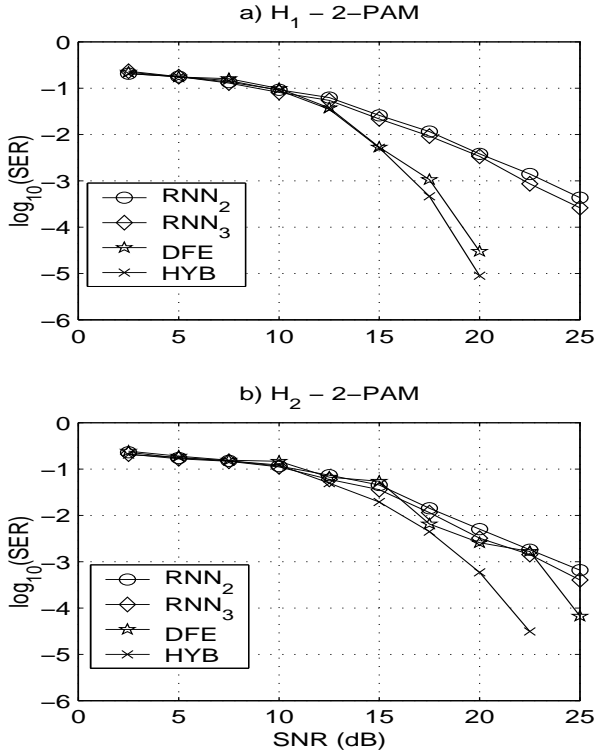


Figure 4: Plot of decimal logarithm of the SER considering  $\tau_d = 50$ , 2-PAM, equalizers configurations shown in Table 3 and channels a)  $\mathcal{H}_1$  and b)  $\mathcal{H}_2$ .

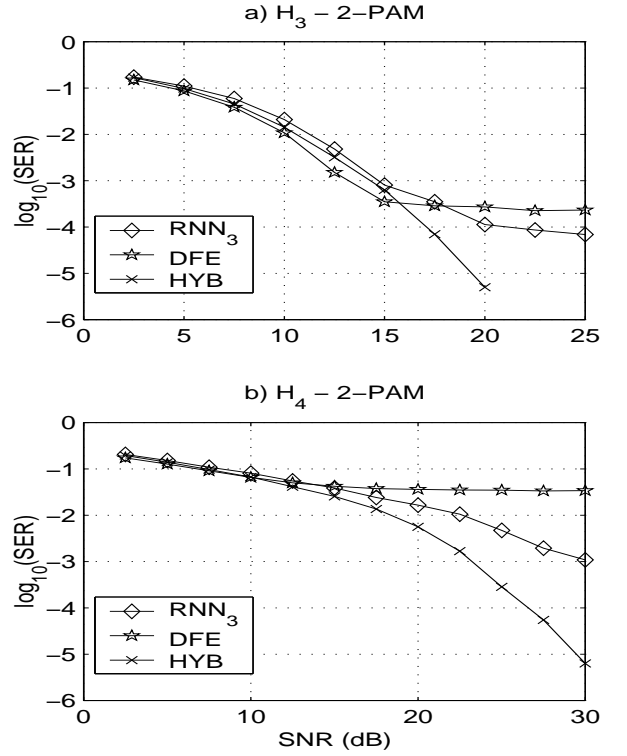


Figure 6: Plot of logarithm of the SER considering  $\tau_d = 3$ , 2-QAM, equalizers configurations shown in Table 3 and channels a)  $\mathcal{H}_3$  e b)  $\mathcal{H}_4$ .

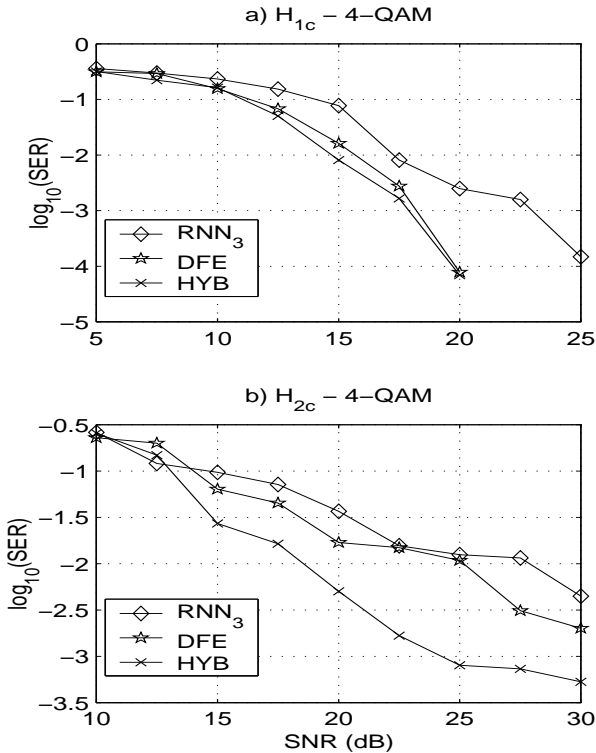


Figure 5: Plot of logarithm of the SER considering  $\tau_d = 50$ , 4-QAM, equalizers configurations shown in Table 3 and channels a)  $\mathcal{H}_{1c}$  and b)  $\mathcal{H}_{2c}$ .

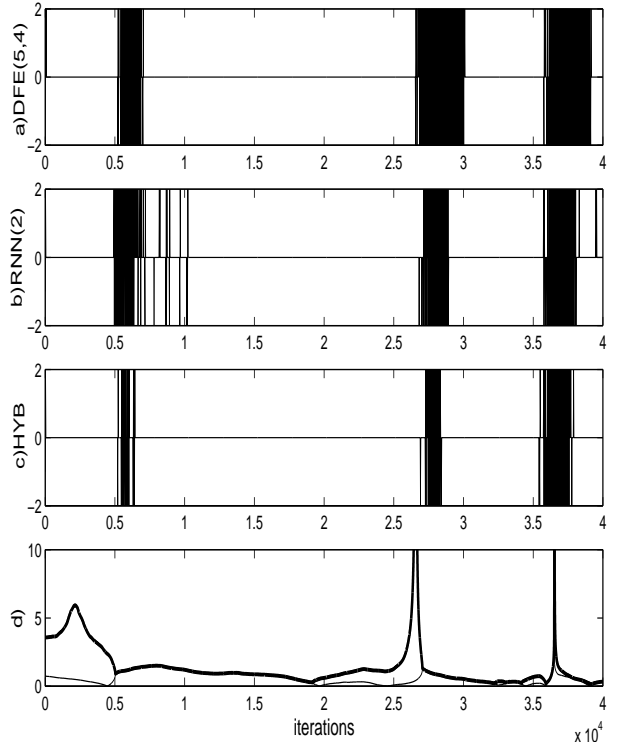


Figure 7: a), b) e c) Output errors considering the equalizers of Table 3 and channel  $\mathcal{H}_5$  d) Absolute root values of the polynomial  $h_0(n)x^2 + h_1(n)x + h_2(n)$  corresponding to channel  $\mathcal{H}_5$ .