Symbol Decision Equalizer using a Radial Basis Functions Neural Network

CORINA BOTOCA, GEORGETA BUDURA
Communications Department
"Politehnica" University of Timişoara
Bd. Vasile .Pârvan, no. 2, 300223, Timişoara
ROMANIA
http://www.utt.ro/english/index.php

Abstract: - This paper presents the problem of multiple quadrature amplitude modulated signals equalization and argues the use of a radial basis functions neural network (RBF-NN) equalizer. Different competitive learning algorithms for the RBF-NN centres determination are discussed. A new competitive learning algorithm is introduced, the rival penalized competitive learning, which rewards the winner and penalizes its first rival. The results of simulations performed in different conditions, are presented showing that the performance of the RBF-NN equalizer, which is based on this new algorithm, is better if compared with other competitive algorithms.

Key-Words: - communication channels, complex equalizer, quadrature amplitude modulated signals, radial basis functions neural network, competitive learning algorithms, centres vectors

1 Introduction

The communication channels are nonlinear and variable in time, in the most general case , due to the intrinsic channels nonlinearities, to the transmission atmospheric conditions, to the man-made noise and to the thermal noise generated by electronic devices. As effect, the transmitted symbols might be affected from nonlinear distortions, fading, additive noise, intersymbol interference (ISI), adjacent channel interference, co channel interference, etc. The technique of reconstructing the transmitted symbols is the equalization.

Traditional approaches to channel equalization are based on the inversion of the global channel response. In digital systems the complete channel inversion is neither required nor desirable. However, traditional techniques requiring the traffic model are obsolete in the context of modern communications, where a mathematical model is not always possible to be drawn. Since transmitted symbols belong to a discrete alphabet, symbol demodulation can be reformulated as a classification problem in the space of the received symbols. Neural networks (NN) are promising candidates, not only because they can learn an arbitrarily nonlinear input output function from examples, but also due to their adaptability, flexibility and outstanding processing speed.

The studies performed during the last decade have already established the superiority of neural equalizers comparative to the traditional equalizers, in conditions of nonlinear distortions and high speed transmission. The multiple quadrature amplitude modulated (M-QAM) signals, more efficient in transmission from the spectral point of view, have known an expanding research interest. The M-QAM signals are severely affected by the nonlinear distortions, because they have a variable envelope modulation. To compensate these unwanted distortions equalizers for complex signals have been developed. The complex NN equalizers are straightforward extensions from the real counterparts [1], obtained by replacing the relevant parameters with complex values.

Various neural equalizers have been developed, mostly combinations between a conventional linear transversal filter (LTF) and a NN which may be a multilayer perceptron (MLP) [1], [2], [3], [4], a RBF-NN [5], [6], [7], [8], [9], [10], or a recurrent neural network [11], [12], [13], [14]. The LTF eliminates the linear distortions, such as ISI, so the NN has to compensate the nonlinearities.

Recently, the RBF-NN have received considerable attention, since the MLP network is plagued by long training times and may be trapped in bad local minima. The RBF-NN is able to approximate any arbitrary nonlinear function in the complex multi-dimensional space with a reduced calculus complexity comparative to other NN. The RBF-NN often provides a faster and more robust solution to the equalization problem than the MLP [1]. In addition, the RBF-NN equalizer has a structure similar to the optimal Bayesian symbol decision equalizer, so its performance is better than the MLP equalizer performance.

Several learning algorithms have been proposed to update the RBF-NN parameters. Usually it is used an unsupervised learning algorithm for the hidden neurons centres determination and a supervised learning algorithm for the output neurons weights calculation [1], [5], [6], [8], [9]. The equalizer performance is directly related to the estimations of the centres. Attractive solutions to the RBF-NN centres determination offer the competitive learning algorithms, because of their simplicity and performance [2].

The typical competitive learning algorithm is the k-means algorithm (KM) [15], which partitions the input data set into k categories (called clusters), each finally represented by its centre. The centres adaptively change starting from some initial values named seed points. The algorithm computes a distance between the input vector and the centres, chooses the winning centre, being the one having the minimum distance and moves it closer to the input vector. The major deficiency of the KM algorithm is that it needs to know the exact number of the clusters k, otherwise it has a poor clustering performance. Unfortunately, in most of the applications, it is hard to determine k in advance. Another drawback of the KM algorithm is the dependence of the classification on several parameters, such as: the seed points, the type of the chosen distance, the number of classes. The k-means algorithm also has the "dead neurons" problem, which means that if a centre is inappropriately chosen, it may never be updated, thus it may never represent a class.

To circumvent the "dead neurons" problem it was proposed an extension of KM algorithm, named the frequency sensitive competitive learning (FSCL) [16]. The chance of a centre to win the competition is indirectly proportional with a parameter named relative winning frequency or "conscience". Although the FSCL algorithm can almost successfully assign one or more seed points to a cluster without the "dead neurons" problem, it also needs to know the exact number of the clusters.

In this paper we introduce a new competitive method to update the centres of a complex RBF-NN equalizer, the rival penalized competitive learning (RPCL) algorithm. This performs appropriate clustering without knowing the number of the clusters, by automatically driving the extra number of seed points far away from the data set. The RPCL algorithm rewards the winning neuron and penalizes with a de-learning rate the second winner, named rival. The algorithm is rather sensitive to the selection of the de-learning rate, but it is quite simple and provides a better convergence than the KM and FSCL algorithms. It also eliminates the "dead neurons" problem. The performance of the RBF-NN equalizer based on the RPCL algorithm is similar to other RBF-NN equalizers reported in literature [9], in the same conditions.

2 The Equalization Problem

Fig. 1 represents a model of the communication system.

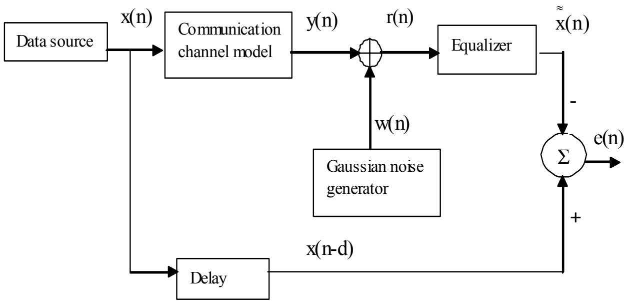

Fig. 1 A model of the communication system

The complex-valued digital sequence x(n) is transmitted through a dispersive complex channel. If the input signal is 4 QAM , its constellation is given by the following relation:

$$x(n) = x_R + jx_I = \begin{cases} x^{(1)} = \ \ 1 + j \\ x^{(2)} = -1 + j \\ x^{(3)} = \ \ 1 - j \\ x^{(4)} = -1 - j \end{cases} \tag{1}$$

The communication channel, that introduces linear and nonlinear distortions, may be modeled as presented in Fig. 2.

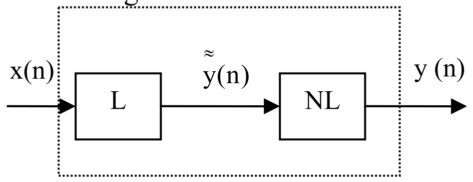

Fig. 2 The nonlinear channel model

Various models with different linear part (L) and nonlinear part (NL) are mentioned in literature [1], [5], [6]. The linear part of the channel is usually a transversal filter with finite response, whose output is given by the relation:

$$\overset{\approx}{y}(n) = \sum_{i=0}^{M-1} a_i x(n-i) \tag{2}$$

Where x(n) is the input signal, n is the discrete time, a_i are the filter coefficients and M the filter order. As example, the model introduced in reference [6], generates the output signal $\overset{\approx}{y}(n)$ according to the relation:

$$\begin{aligned}\overset{\approx}{y}(n) = &(0.7409 - 0.7406\,j)x(n) - \\ &-(0.8890 - 0.2961\,j)x(n-1) + \\ &+(0.1556 - 0.0223\,j)x(n-2)\end{aligned} \tag{3}$$

The nonlinear part produces at the channel output:

$$y(n) = \overset{\approx}{y}(n) - 0.055[\overset{\approx}{y}(n)]^2 + 0.14[\overset{\approx}{y}(n)]^3 \tag{4}$$

The channel output y(n) is corrupted by adding a complex-valued noise w(n), usually a white Gaussian noise with a zero mean and a dispersion of σ^2, so the received signal is:

$$r(n) = y(n) + w(n) \tag{5}$$

In experiments, the signals x(n) and w(n) are assumed uncorrelated. The real and imaginary part of noise, $w_R(n)$ respectively $w_I(n)$ are also assumed mutually independent.

The task of the equalizer is to reconstruct the transmitted symbols as accurately as possible, producing an estimation $\overset{\approx}{x}(n)$, based on the noisy received signal r(n) and the desired delayed signal x(n-d). From the NN point of view, the objective of equalization is the separation of the received symbols in the output signal space, whose optimal decision boundaries are generally highly nonlinear.

3 The Equalizer Architecture and Training Algorithms

3.1 The RBF Neural Network

The channel output vector is passed through the RBF-NN equalizer, consisting of a LTF of order m and a RBF-NN, as presented in Fig. 3. The received signal r(n) applied as input to the RBF-NN input is the sequence r(n)=[r(n) r(n-1) ...r(n-m+1)]^T. Because it involves m terms of the delayed version of the received signal, there are $N_S=4^{M+m-1}$ possible combinations of the channel input sequences (M is the channel model order), i.e

x(n)=[x(n)x(n-1)…x(n-m-M+2)]^T. Correspondingly, the noise-free channel output vector is y(n)=[y(n) y(n-1) …y(n-m+1)]^T and it has also N_S desired corresponding states. As depicted in Fig. 3, the RBF has two layers, the hidden layer and the output layer. The RBF-NN input and output are both complex and the nonlinearity of hidden neurons is a real function. The real and the imaginary part of the signals are treated separately, in the same manner. The hidden layer is composed of an array of computing neurons, each one having a vector parameter c, called centre. Each neuron computes a distance between its centre and the network input vector. This distance may be of different types, but usually a Euclidian norm is used, and is divided by a parameter ρ, called width or radius, which is the spread of the corresponding centre.

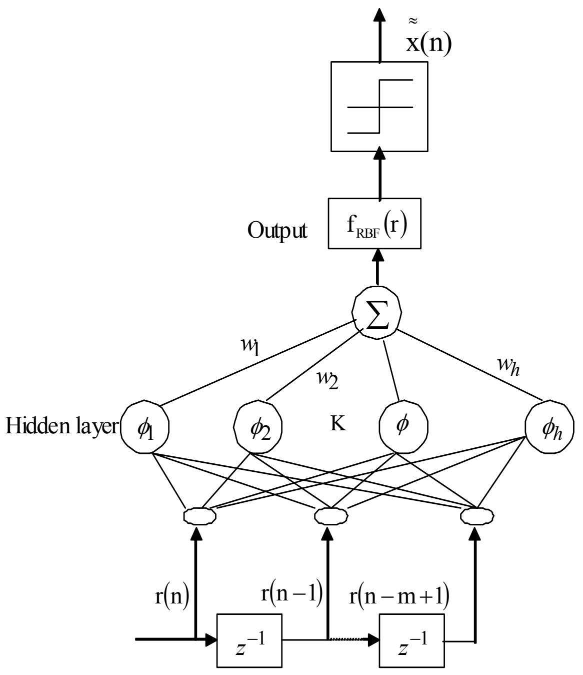

Fig3. The RBF neural network

The result is passed through a real, nonlinear activation function $\phi_i(\bullet,\rho)$

$$\phi_i = \phi\left((r(n)-c_i(n))^H (r(n)-c_i(n))/\rho\right) \quad 1 \le i \le n_h \tag{6}$$

where r(n) is the complex input vector, $c_i(n)$ is the i centre vector, which is also a complex vector, ρ is the centre spread parameter, n_h is the number of hidden neurons.The operator $(\bullet)^H=((\bullet)^T)^*$, where $(\bullet)^T$ is the transposition operator and $(\bullet)^*$ is the complex conjugation operator.

The nonlinear output function is usually the Gaussian function:

$$\phi(\chi^2,\rho) = e^{-\frac{\chi^2}{\rho}} \tag{7}$$

Similarity with the Bayesian equalizer imposes that the spread parameter $\rho=2\sigma^2$ [6] ,where σ^2 is the noise dispersion given by relation:

$$\sigma^2 = E[\|r(n)-c_i(n)\|^2] \tag{8}$$

where E is the is the second order momentum and $\|\ \|$ is the Euclidian norm.

In the output layer there are two neurons for each 4-MAQ signal class, one for the real part and another for the imaginary part, which calculate a linear function:

$$f_{RBF}(r) = \sum_{i=1}^{n_h} \phi_i w_i \tag{9}$$

where w_i are the complex weights. According to the relation (8), f_{RBF} becomes:

$$f_{RBF}(r) = \sum_{i=1}^{n_h} w_i e^{-\frac{(r(n)-c_i(n))^H (r(n)-c_i(n))}{\rho_i}} \tag{10}$$

3.2 The Rival Penalized Competitive Learning Algorithm

The RPCL algorithm is a variant of the FSCL algorithm and it performs appropriate clustering without knowing the number of the clusters. It determines not only the winning centre j with relation (11) , the one having a minimum distance to the input, but also the second winning centre r, named rival:

$$j = \arg\min(\gamma_i \|r(n) - c_i(n)\|), \quad i = \overline{1, n_h} \tag{11}$$

$$r = \arg\min\big|_{i \neq j} (\gamma_i \|r(n) - c_i(n)\|), \quad i = \overline{1, n_h} \tag{12}$$

where r(n) is the input vector, $c_i(n)$ is the i centre vector, γ_i is the relative winning frequency of the centre i, dependent on the number of times s_i when the centre i has won the competition in the past

$$\gamma_i = \frac{s_i}{\sum_{i=1}^{n_h} s_i} \tag{13}$$

Alike in the FSCL algorithm, if a center has won too often the competition "it feels guilty" and it pulls itself out of the competition. The centers that have won the competition during the past have a reduced chance to win again, inverse proportional with their relative winning frequency term γ.

The winning centre is moved with a fraction η, $\eta \in (0,1)$, towards the input and the rival is moved away from the input with a ratio β, $\beta \in (0,1)$ called the de-learning rate. So the learning law can be synthesized as follows:

$$\boldsymbol{c_i(n+1)} = \begin{cases} \boldsymbol{c_i(n)} + \eta\, [r(\boldsymbol{n}) - \boldsymbol{c_i(n)}] & if\ i = j \\ \boldsymbol{c_i(n)} - \beta\, [r(\boldsymbol{n}) - \boldsymbol{c_i(n)}] & if\ i = r \\ \boldsymbol{c_i(n)} \quad if\ i \neq j \quad and \quad i \neq r \end{cases} \tag{14}$$

If the learning speed η is chosen much greater than β, with at least one order of magnitude, the number of the data clusters will be automatically found. In other words, if the number of clusters is unknown and the number of the centres n_h is greater than the number of the clusters, than the centres vectors will converge towards the middle point of the input data classes. If n_h is smaller than the number of input data clusters, than the network will oscillate during training, indicating that the number of the centres must be increased.

So the RPCL algorithm moves away the rival, in each iteration, converging much faster than the KM and the FSCL algorithms. The extra number of the seed points, respectively the difference between n_h and the number k of the clusters will be driven away from the data set.

3.3 The Least Mean Square Algorithm

A supervised algorithm may be used to update the output neurons weights, for instance, the least mean square algorithm, given by the following relations:

$$w_i(n+1) = w_i(n) + \alpha e(n) \phi(n) \tag{15}$$

where α is the learning constant, chosen in the interval (0,1) and e(n) is the complex error, determined with the relation:

$$e(n) = x(n-d) - f_{RBF}(r) \tag{16}$$

This algorithm minimizes the mean square error (MSE):

$$MSE = \frac{1}{N} \sum_{i=1}^{N} e_i^2(n) \tag{17}$$

where N is the number of input sequences.

4 Simulation Results

The RBF-NN equalizer was tested in Matlab, using the KM, FSCL and RPCL algorithms, in different conditions of noise, for different orders of LTF and delays d. There were generated 4-QAM signals, using an uniform distribution, independently the real part from the imaginary part. Simulations were done using the channel model introduced in reference [6] and presented in section 2. A white noise w(n) was generated and added to y(n). The number of the hidden neurons, respectively of the desired centres, was chosen equal to N_S, the number of possible states of y(n). For M=3, m=1 N_S results 64. For the RBF-NN output layer there were used 8 neurons, one for the real part and another for the imaginary part of each of the four possible states of the 4-QAM signal. The centre spread was chosen to 0.28. There were applied N=7000 input signal sequences x(n), x(n)=[x(n) x(n-1) x(n-2)] to train the RBF-NN centres.

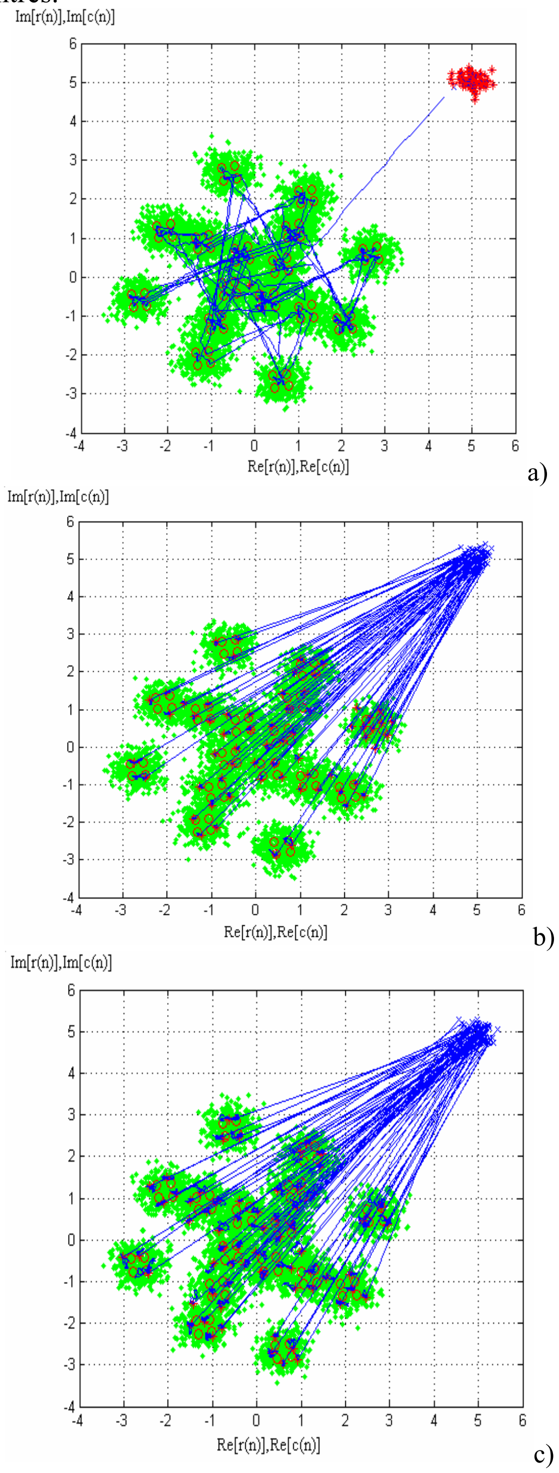

Fig.4 The output channel signals, the corrupted received signals r(n), the initial and found positions of the RBF-NN centres c(n), in case of a SNR=13dB, after 100 training iterations: a) using the KM algorithm; b) using the FSCL algorithm; c) using RPCL algorithm; (Legend: "o" - the output channel signals; "*"- the corrupted received signals; "×" - initial centres positions; "+" - the found centres; "–" evolution of the centres)

The were chosen 70 seed points randomly initialized around (5, 5j) as it can be seen in Fig.4 a), b) and c). The best results were obtained for the following learning constants: η=0.05, β=0.0001 and α=0.01. Fig.4 a), b) and c) represent the output channel signals y(n), the corrupted received signal r(n), the initial and final positions of the RBF-NN centres c(n) in the case of a SNR=13dB (σ^2=0.05), after 100 training iterations, using the KM, FSCL and RPCL algorithms. The KM algorithm failed to find the desired centres, because of the "dead neurons" problem. The FSCL algorithm didn't find all the desired centres. The RPCL algorithm, succeed to orientate the RBF-NN centres to the desired free of noise output channel signals. In addition, the RPCL algorithm has driven away the extra number of seed points. As one can observe the RPCL algorithm could also find closer positions of RBF-NN centres to the desired output channel signals than the FSCL algorithm, so its convergence is better.

Fig. 5 depicts the MSE evolution, using the new RPCL algorithm, during 3000 epochs for different signal to noise ratios (SNR=10dB and SNR=5 dB) and order m (m=1 and m=2) of the LTF and a delay of d=1. This performance is similar to the MLP NN equalizers, in conditions of a lower computational cost.

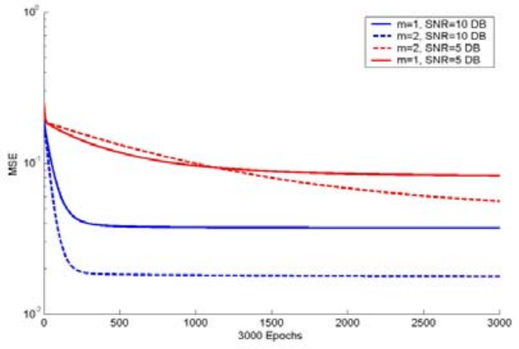

Fig.5 The comparative evolution of the MSE during 5000 epochs for a SNR =10 dB and SNR =5 dB, m=1 and m=2 (solid line - FSCL algorithm; dotted line - RPCL algorithm)

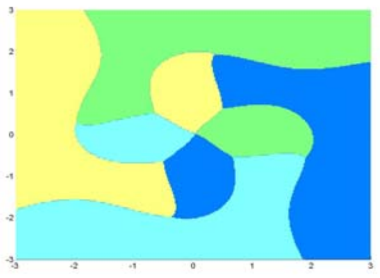
Fig.6 The output signals space partition

To represent the decision regions of the RBF-NN complex equalizer, the complex output space was divided using a sampling step of δ=0.02. Fig. 6 represents the output signals space partition, which has strong nonlinear decisions boundaries.

5 Conclusions

The main drawback of the neural network equalizers is the computational complexity and the extensive training. The proposed competitive algorithm, namely RPCL algorithm, while training the centres of the RBF-NN network, rewards the winner and penalizes its closest rival. It is rather simple, generates strong nonlinear regions of decision in the signal space, yet having a better and faster convergence. In comparison with the classic KM algorithm, it doesn’t have the "dead neurons" problem. If compared with the FSCL algorithm, it has a better and faster convergence. So the RPCL algorithm is adequate to the adaptive equalization of fast varying signals corrupted with strong linear and nonlinear distortions. Because of its structure, similar to the Bayesian equalizer, the performance of the RBF-NN equalizer is superior to the LTF and MLP equalizers. The MSE performance of the RPCL equalizer is similar to others RBF-NN equalizers reported in literature, if tested in the same conditions. In order to improve the equalizers performances the order of the LTF filter coupled with the RBF-NN neural network should be increased or the feedback should be introduced.

References:

[1] M. Ibnkahla, Applications of Neural Networks to Digital Communication a Survey, IEEE *Signal Processing Magazine*, November, 1997, pp.1186-1215

[2] S.Bouchired, M. Ibnkahla, Décision Neuronale Appliquée a L’égalisation De Canaux Satellitaires Mobiles , Dix-septième colloque , Gretsi, Proceedings, 1999,pp.1125-1128

[3] T. Kim and T. Adali, Fully complex multi-layer perceptron network for nonlinear signal processing, *Journal of VLSI Signal Processing*, 32, 1, Aug./Sep, 2002, pp. 29-43

[4] S.Sezer, Ph.Power, Neural Equalization Filtering for Highbandwidth QAM Channels, IEEE ICT 2001, Proceedings, vol.2, 2001, pp.225-229

[5] I.Cha, S.Kassam, Channel Equalization Using Complex-Valued Radial Basis Function Networks, *IEEE. Journal Select. Areas Commun,*.13,iss. 1, 1995, pp.122-131

[6] S. Chen, S. McLaughlin, B. Mulgrew, Complex-Valued Radial Basis Function Network. Network Architecture and Learning Algorithms, *Signal Processing,* 35, 1994, pp. 19-31

[7] S. Chen, et al., Complex-Valued Radial Basis Function Network. Part II: Application to Digital Communications Channel Equalization, *Signal Processing*, 36, 1994, pp. 175-188

[8] Q. Gan, P. Saratchandran, N. Sundararajan, K. R. Subramanian, A Complex Valued Radial Basis Function Network for Equalization of Fast Time Varying Channels, *IEEE Trans. On Neural Networks*, 10, issue 4, 1999, pp. 958-960

[9] D.Jianping, N.Sundararajan, P. Saratchandran, Communication Channel Equalization Using Complex-Valued Minimal Radial Basis Function Neural Network, *IEEE Trans. On Neural Networks,13*, *No.3*, 2002, pp.687-696

[10] J. Lee, C. D. Beach, N. Tepedelenlioglu, Channel Equalization Using Radial Basis Function Network, IEEE Int. Conf. On Neural Networks, vol. 4, 1996, pp. 1924-1928

[11] M.T.Madeira da Silva, M. Gerken, A RNN-LC Hybrid Equalizer, XI European Signal Processing Conference, Toulouse, France, Proceedings, 2002, pp.341-344

[12] J.Choi, M. Bouchard, T.H.Yeap, Decision Feedback Recurrent Neural Network Equalization with Fast Convergence Rate, I*EEE Trans. On Neural Networks,* vol. 16, n. 3, May 2005, pp. 699-708

[13] G. Kechriotis, E. Zervas, E. S. Manolakos, Using Recurrent Neural Networks for Adaptive Communication Channel Equalization, *IEEE Trans on Neural Networks*, 5, 1994, pp.267-278

[14] R. Parisi, E. Di Claudio, G. Orlandi, B. Rao, Fast Adaptive Digital Equalization by Recurrent Neural Networks, *IEEE Trans. Signal Process.*, 45, Nov., 1997, pp.2731-2739

[15] Hecht-Nielsen, *Neurocomputing,* Addison-Wesley Publishing Company, pp.63-74, New York, 1990

[16] S.C.Ahalt, A.K.Krishnamurty, P.Chen, D.E.Melton, Competitive Algorithms for Vector Quantization, *Neural Networks*, 3, 1990, pp.277-291