16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, August 25-29, 2008, copyright by EURASIP

LEVENBERG-MARQUARDT LEARNING NEURAL NETWORK FOR ADAPTIVE PRE-DISTORTION FOR TIME-VARYING HPA WITH MEMORY IN OFDM SYSTEMS

Rafik Zayani[1], Ridha Bouallegue[1], Daniel Roviras[2]

[1]6'Tel Unit research / SUP'COM, Tunis, Tunisia
[2]IRIT Laboratory / ENSEEIHT, Toulouse, France

ABSTRACT

This paper presents a new adaptive pre-distortion (PD) technique, based on neural networks (NN) with tap delay line for linearization of High Power Amplifier (HPA) exhibiting memory effects. The adaptation, based on iterative algorithm, is derived from direct learning for the NN PD. Equally important, the paper puts forward the studies concerning the application of different NN learning algorithms in order to determine the most adequate for this NN PD. This comparison examined through computer simulation for 64 carriers and 16-QAM OFDM system, is based on some quality measure (Mean Square Error), the required training time to reach a particular quality level and computation complexity. The chosen adaptive pre-distortion (NN structure associated with an adaptive algorithm) have a low complexity, fast convergence and best performance.

1. INTRODUCTION

Orthogonal frequency division multiplexing (OFDM) was initially presented in 1966. It has been used in the digital terrestrial television broadcasting and the wireless local area network. Hence, OFDM has received much attention in the development of the fourth generation mobile communication systems in recent years [3]. However, OFDM exhibits large peak-to-average power ratios, i.e., large fluctuations in their signal envelopes. Indeed, the performance of the transceivers is very sensitive to nonlinear distortions caused by the high power amplifier (HPA). Among all linearization techniques, digital pre-distortion is one of the most cost effective and its principle is to distort the HPA input signal by an additional device called a pre-distorter which characteristics are the inverse of those of the amplifier. In reality, the power amplifier characteristics may change over time because of temperature drift, component aging, power level, biasing variations, frequency changes, etc. Thus, it is desirable to make an adaptive pre-distortion and that is why we are focused on the adaptation of pre-distorter characteristic.

Systems which use OFDM as modulation scheme, memory effects of high power amplifiers cannot be ignored due to the broadband input signal. These memory effects may be explained by frequency dependence of components or by thermal phenomena [4].

The aim of our paper is to check the possibility of the application of a neural network to perform the function of the HPA pre-distorter of the OFDM signals. It seems that neural networks, which are nonlinear in their nature, could be a good tool to compensate for nonlinearity. Additionally, their regular structure is well fitted to an efficient implementation. Indeed in [1], the authors present a preliminary implementation of a data pre-distortion system using a multilayer perceptron neural network which forms an adaptive nonlinear device whose response can approximate inverse transfer functions of time-varying HPA nonlinearities.

In this work we extend this solution to a one capable to compensate not only for the nonlinearities and their time-varying characteristics but also for the memory effects in the HPA. The adaptation, based on iterative algorithm, is derived from direct learning for the NN PD, the crucial point then is to find a suitable training algorithm able to cope with the described network and the training data set. In short, this paper compares the performance of five neural network learning algorithms in order to determine the most adequate for this adaptive NN PD.

The NN techniques used are the Gradient Descent backpropagation (GD), the Gradient Descent backpropagation with the momentum (GDm), the Conjugate Gradient BP (CGF), the Quasi-Newton method (BFG) and the Levenberg-Marquardt (LM). This comparison is carried out for 64 carriers and 16-QAM OFDM system with a saleh's TWT amplifier, is based on some quality measure (Mean Square Error), the required training time to reach a particular quality level and computation complexity.

The paper is organized as follow. Section II describes the proposed system scheme with neural network pre-distorter. A review of potential MLP training algorithms is presented in section III. Section IV presents comparison results of the five backpropagation methods proposed for adaptive PD in terms of performance and complexity and shows performance results of the chosen PD technique in compensating distortions of HPA with memory. The conclusion is given in section V.

2. SYSTEM DESCRIPTION

Figure 1 shows the baseband discrete equivalent communication system model for OFDM system with pre-distortion, where c_k is the k-th transmitted symbol, which is mapped in 16-QAM, x_n is the n-th transmitted OFDM sample, y_n is the same sample at the output of the pre-distorter and z_n denotes the amplified one.

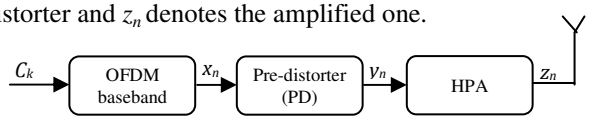

Figure 1 - Simplified OFDM transmitter with PD and HPA

The pre-distorter (PD) of figure 1 is an adaptive nonlinear device with memory that precomputes and cancels all distortions caused by HPA.

2.1 HPA model

For the HPA model with memory, we have considered a Hammerstein system (see figure 2) which it can be represented by a time-varying memoryless HPA followed by a linear filter.

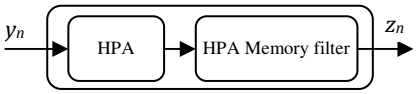

Figure 2 - Model of the HPA with memory

For the nonlinear part of the HPA, we have chosen Saleh's well-established TWTA model [7]. In this model, AM/AM (amplitude modulation to amplitude modulation) and AM/PM (amplitude modulation to phase modulation) conversion can be represented as follow:

$$A(r) = \frac{\alpha_a r}{1+\beta_a r^2} \quad \text{and} \quad P(r) = \frac{\alpha_p r^2}{1+\beta_p r^2} \qquad (1)$$

where r is the input modulus of the TWTA and α_a, β_a , α_p, β_p are four adjustable parameters. The output of the TWTA can be represented as:

$$z(t) = A(r)\exp\left(j.(\omega_c t + \emptyset(t) + P(r))\right) \qquad (2)$$

where $\emptyset(t)$ is the phase of the input signal.

As a non-stationary (time-varying) model, we consider the memoryless model where the four parameters α_a, β_a , α_p, β_p are changing with time according to the following conditions [1]: $1.5 \leq \alpha_a \leq 3$, $0.5 \leq \beta_a \leq 2$, $2 \leq \alpha_p \leq 4$ and $7 \leq \beta_p \leq 9$. The following figures represent the variation of AM/AM and AM/PM in order to show the extent of the HPA variations used in this work.

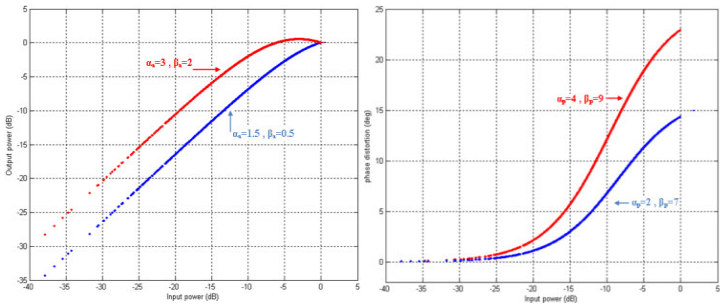

Figure 3 - AM/AM and AM/PM characteristic variations

The linear subsystem in the amplifier that captures the memory effects is modeled by a low pass filter.

2.2 Adaptive Pre-distortion for HPA

The aim of our investigation was to apply a simple neural network to perform the function of the HPA pre-distorter of the OFDM signal. As we mentioned earlier, the adaptation property of the pre-distorter is a very desired feature because the characteristics of power amplifiers are time variant.

Then, the pre-distortion architecture presented here is basically derived from a post-distortion adaptive structure which may employ two general alternatives for its operation. These alternatives are:

1st Alt: Loading the pre-distorter with completely trained coefficients after a complete learning stage. (PD is here stationary (Figure 4)).

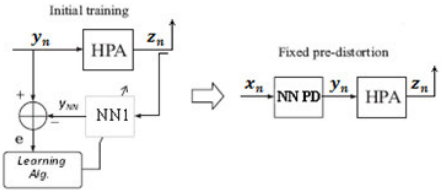

Figure 4 - Block diagram for training of the PD with HPA

2nd Alt: Simultaneous updating of the pre-distorter during the adaptation at the post-distortion loop. (PD is here adaptive (Figure 5))

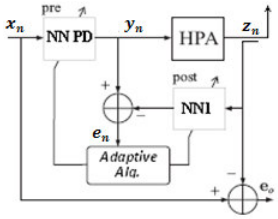

Figure 5 - Simultaneous PD updating

Figure 5 shows the detailed scheme of an adaptive pre-distortion system based on feed-forward neural network. x_n denotes the input signal to the pre-distorter, y_n denotes the output signal from the pre-distorter witch is sent as input to the HPA and z_n denotes the HPA output signal.

The weights of the neural network pre-distorter (NN PD) are determined by copying the weights of NN1. These weights are adjusted using an adaptive algorithm.

2.3 Applied neural network structure

The pre-distorter used in this paper is a neural network mimetic structure (figure 6), which is composed of a Linear Neural network (LN), with 4 memory cells (as a linear filter with 4 poles) followed by a memoryless Nonlinear Neural network (NLN), with one hidden layer with nine neurons (with sigmoid activation function) and two linear neurons in the output layer. Using this mimetic scheme (LN-NLN), we realize separately the memory pre-distortion with the linear network and the pre-distortion of the memoryless HPA nonlinearities with the nonlinear neural network.

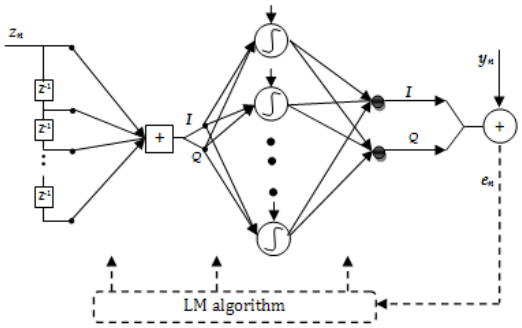

Figure 6 - Linear Network LN + Non-Linear Network NLN pre-distorter structure

It is well known that each neuron in the network is composed of a linear combiner and an activation function which gives the neuron output:

$$y_{lj} = f\left(\sum_{i=0}^{N_{l-1}} w_{l,j,i} x_{l-1,i} + b_{lj}\right) \qquad (3)$$

where $w_{l,j,i}$ is the weight which connects the i-th neuron in layer l-1 to the j-th neuron in layer l, b_{lj} is the bias term and $x_{l-1,i}$ denotes the i-th component of the input signal to the neuron.

3. TRAINING ALGORITHMS

In this section, we review different algorithms used in this investigation to train the neural network pre-distorter: Gradient descent backpropagation (GD), Gradient descent backpropagation with the momentum (GDm), Conjugate Gradient BP (CGF), Quasi-Newton (BFG) and Levenberg-Marquardt (LM).

• *Gradient Descent BP (GD)*

The gradient based methods are the most straightforward training algorithms for feed forward multilayer perceptron networks [5] and there are two different methods in which this gradient descent algorithm can be implemented: incremental mode and batch mode. The simplest implementation of back propagation learning updates the network weights and biases in the direction in which the performance function decreases more rapidly. The new weight vector x_{k+1} can be adjusted as:

$$x_{k+1} = x_k - \mu g_k \quad (4)$$

where x_k is the vector of current weights and biases, μ is the learning rate and g_k is the gradient of the error with respect to the weight vector. The computation of g_k is presented in [5].The negative sign indicates that the new weight vector $x_{k+1}$is moving in a direction opposite to that of the gradient.

• *Gradient Descent BP with momentum (GDm)*

The convergence of the network by backpropagation is a crucial problem because it requires many iterations. To mitigate this problem, a parameter, called "Momentum", can be added to BP learning method by making weight changes equal to the sum of fraction of the last weight change and the new change suggested by the gradient descent BP rule (Eq. 6, [5]). The momentum is an effective means not only to accelerate the training but also to allow the network to respond to the (local) gradient.

Then, the new weight vector is adjusted as [5]:

$$x_{k+1} = x_k - \mu g_k + \alpha(x_k - x_{k-1}) \quad (5)$$

where the parameter α is the momentum constant, which can be any number between 0 and 1.

• *Conjugate gradient BP (CGF)*

The standard backpropagation algorithm adjusts the weights in the steepest descent direction, which does not necessarily produce the fastest convergence [6]. And it is also very sensitive to the chosen learning rate, which may cause an unstable result or a long-time convergence [4]. As a matter of fact, several conjugate gradient algorithms have recently been introduced as learning algorithms in neural networks [5]. They use, at each iteration of the algorithm, different search directions in a way which produce generally faster convergence than steepest descent directions [2]. In the conjugate gradient algorithms, the step size is adjusted at each iteration. The conjugate gradient used here is proposed by Fletcher and Reeves [5][6].

All conjugate gradient algorithms start out by searching in the steepest descent direction on the first iteration.

$$p_0 = -g_0 \quad (6)$$

The search direction at each iteration is determined by updating the weight vector as:

$$x_{k+1} = x_k + \mu_k p_k \quad (7)$$

where:

$$p_k = -g_k + \beta_k p_{k-1} \quad (8)$$

For the Fletcher-Reeves update, the constant β_k is computed by:

$$\beta_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}} \quad (9)$$

This is the ratio of the norm squared of the current gradient to the norm squared of the previous gradient.

• *BFGS Quasi-Newton (BFG)*

In Newton methods the update step is adjusted as:

$$x_{k+1} = x_k - H_k^{-1} g_k \quad (10)$$

where H_k is the Hessian matrix (second derivatives) of the performance index at current values of the weights and biases.

Newton's methods often converge faster than conjugate gradient methods. Unfortunately, they are computationally very expensive, due to the extensive computation of the Hessian matrix H coming along with the second-order derivatives [6]. The quasi-Newton method that has been the most successful in published studies is the Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update [5].

• *Levenberg Marquardt (LM)*

Similarly to quasi-Newton methods, the Levenberg-Marquardt algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. Under the assumption that the error function is some kind of squared sum, then the Hessian matrix can be approximated as:

$$H = J^T J \quad (11)$$

and the gradient can be computed as:

$$g = J^T e \quad (12)$$

where J is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases. The Jacobian matrix determination is less computationally expensive than the Hessian matrix; e is a vector of network errors.

Then the update can be adjusted as:

$$x_{k+1} = x_k - [J^T J + \mu I]^{-1} J^T e \quad (13)$$

The parameter μ is a scalar controlling the behavior of the algorithm. For $\mu = 0$, the algorithm follows Newton's method, using the approximate Hessian matrix. When μ is high, this becomes gradient descent with a small step size.

4. SIMULATION RESULTS AND DISCUSSION

It is very difficult to know which training algorithm will be the fastest and the most adequate for a given problem. It will depend on several factors including the complexity and the type of problem, the data set of the training base, the number of weights and biases in the network and the required training time, hardware resources and mean squared error between the actual and desired network response.

In this section we carry out a certain number of comparisons of the various training algorithms to enhance the learning of the memoryless nonlinear (NLN) part of the mimetic pre-distorter structure used in this investigation (see figure 6). The neural network is of feed-forward type with two inputs, two outputs (I and Q) and a hidden layer of nine neurons (2-9-2). The activation function is sigmoid for hidden layer and linear for output ones. Also, 312 OFDM samples were employed for the learning process.

In this investigation[1], the NLN is employed to approximate inverse transfer functions of the amplifier used in OFDM system with 64 carriers and 16-QAM. Accordingly, the accuracy expected from the approximation can affect the performance of the various algorithms. The following figure plots for each method, the Mean Square Error versus iteration number averaged over 30 simulations. We can see that the MSE in the LM algorithm decreases much more rapidly than the other algorithms.

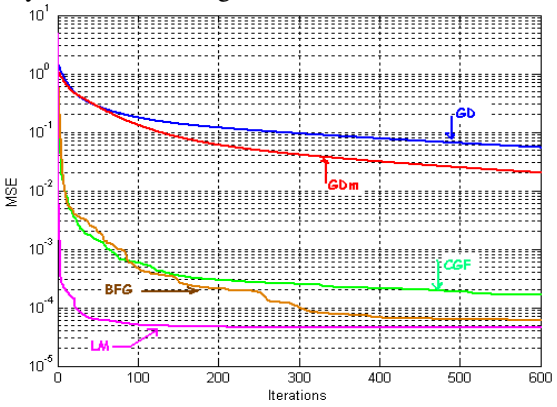

Figure 7- Mean square error versus iteration for different algorithms

At this point, we can say that the LM algorithm gives more accurate results in terms of convergence speed. Nevertheless, it is important to consider the algorithmic complexity. The following table summarizes the results of the comparative study of the five mentioned algorithms in terms of complexity. The variable *Nflops* (number of floating operations) is the number of computations that each method required to run per epoch while *Ntflops* is the number of computation that each method required to reach the minimum MSE. In each case, the network is trained until the squared error is less than 10^{-3}.

For the calculation of the number of floating operations, additions and subtractions are one flop if real and two if complex. Multiplications and divisions count one flop each if the result is real and six flops if it is not.

Algorithm	*Nflops*	*Ntflops*
LM	5973400	1.5651e+007
BFG	402710	2.225e+007
CGF	285300	2.472e+007
GDm	296574	*
GD	197663	*

Table 1. Computation comparison for different algorithms

(* Required training goal was not reached with 2.10^5 Epochs)

As can be seen in Table 1, the Levenberg-Marquardt algorithm is obviously quite well suited for the used neural network training. Although it requires the most significant number of computation per epoch (because of the Hessian computation), it requires the lower amount of computation flop (*Ntflops*) for the mean square error convergence goal.

The following figure indicates the number of computation (*Ntflops*) required to converge versus the Mean Square Error convergence goal. Again, we observe as the error goal is reduced, that the improvement provided by the LM algorithm becomes more pronounced. LM algorithm performs better than other algorithms as the MSE goal is reduced.

[1] *All experiments were carried out Matlab running on HP pavilion ze5500 with a Mobile Intel Pentium IV 2.66 GHz processor and 512 Mo RAM.*

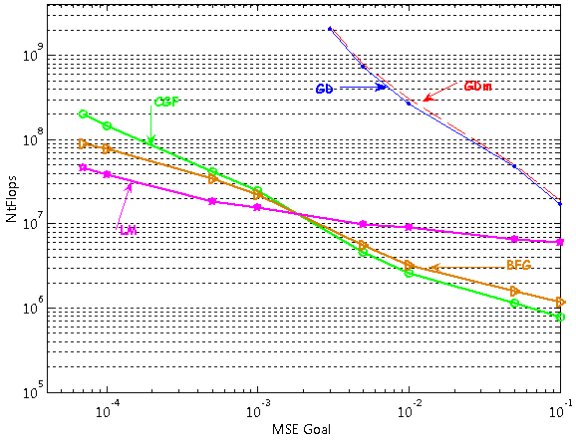

Figure 8 - Computation number required versus mean square error

As we mentioned previously, the HPA can be a time-varying system. In this subsection, we assume that the four parameters α_a, β_a, α_p, β_p are time varying as presented in [1]; Thus, we study the performances of these algorithms for an adaptive pre-distortion (figure 5), in the case of a non-stationary amplifier.

In a first phase, we train the neural network during a "To" time in order to fix the "NN PD" (figure 4). The following figure represents learning curves of the neural network with various algorithms according to time "t" such as (0<t<To).

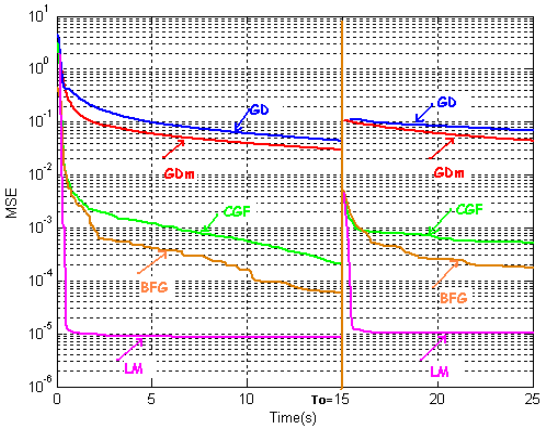

Figure 9 - Mean square error versus time

The amplifier used is a Saleh's TWTA model with a given set of parameters ($\alpha_a = 2$, $\beta_a = 1$, $\alpha_p = 4$, $\beta_p = 9$). We note that all algorithms converge towards different MSE and that LM reaches the smallest.

At time "To", we abruptly change the parameters of the amplifier such as ($\alpha_a = 3$, $\beta_a = 2$, $\alpha_p = 2.5$, $\beta_p = 7.5$) and we compare the convergence of the various algorithms in time. According to figure 9, we clearly notice that the Levenberg-Marquardt algorithm is the fastest and ensures the best convergence towards a minimum error.

We now present the validity of the proposed pre-distortion technique (figure 6) for compensation of HPA nonlinear distortions and their memory effects in the same OFDM system as used previously. The memoryless nonlinear model for HPA selected is the Saleh's TWTA modeled by equation (1) where $\alpha_a=2$, $\beta_a=1$, $\alpha_p=4$ and $\beta_p=9$. The linear subsystem in the HPA that captures the memory effects is implemented using a low pass filter (see response in figure 12) with 3 poles (0.7692, 0.1538, 0.0769) [8]. For the adaptive training algorithm, we have used Levenberg-Marquardt which exhibits the best

performance compared to other training algorithms as we demonstrated previously. Additive White Gaussian Noise (AWGN) channels were assumed to clearly observe the effect of nonlinearity and performance improvement by the proposed PD.

We need a criterion to show how much power back-off is needed for optimum power efficiency. In the simulations, we define the input back-off (IBO) as:

$$IBO = 10log_{10}(\frac{A_0^2}{P_{in}}) \qquad (14)$$

where P_{in} is the average input power and A_0 is the maximum output amplitude .

Symbol Error Rate (SER) diagrams are a typical performance measure for qualifying the compensating ability of proposed pre-distortion structures to reduce HPA distortions. Then, the following figure shows the SER performance versus Signal to Noise Ratio (SNR) in systems with a linear HPA along with nonlinear memory HPA without pre-distortion, with NN memoryless pre-distortion (NLN) and memory pre-distortion (LN-NLN). The realistic level of memoryless nonlinear distortions is considered by working with input back-off (IBO) equal to 7dB.

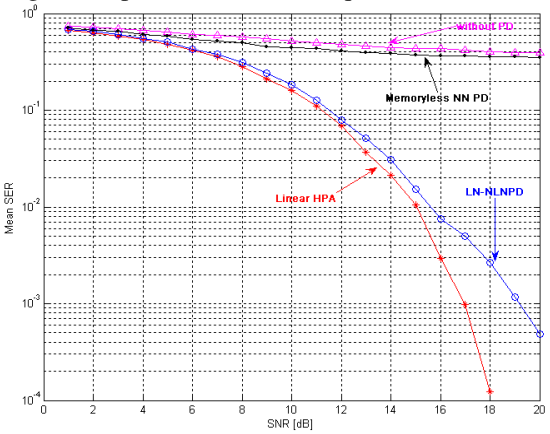

Figure 10 - SER performance for 16-QAM OFDM with 64 carriers at IBO =7dB

Figure 11 represents AM/AM curves of the amplified signal versus input signal without pre-distortion and pre-distorted signal versus input signal for the studied (LN-NLN) pre-distortion.

The memoryless pre-distortion (NLN) is not included in this comparison since it has a lower performance than the LN-NLN pre-distortion.

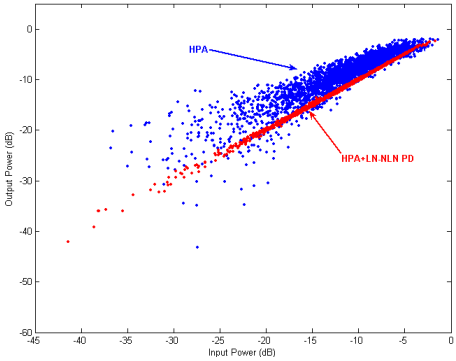

Figure 11 - AM/AM curves for LN-NLN PD

We show on figure 12 the response of both the HPA memory filter and the converged Linear Network (LN). We see that the memory pre-distortion has also been successfully identified by the LN.

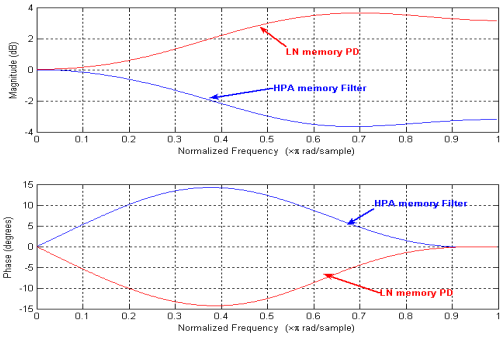

Figure 12 - Memory PD identification

5. CONCLUSION

We put forward an adaptive neural network pre-distorter which can automatically compensate for amplifier nonlinearity and thus makes it possible to transmit OFDM signals without incurring intolerable distortions.

The performances of the proposed neural network pre-distorter depend on the BP training algorithm. This dissertation compares the performance of five neural network training methods in adaptive pre-distortion.

We demonstrated by simulations that the Levenberg-Marquardt algorithm has the fastest convergence in terms of iteration number. In many cases, LM is able to obtain lower mean square errors than any of the other algorithms tested. Also LM requires the lower amount of computation for low MSE. This advantage is mainly noticeable if very accurate quality level is required.

In short, LM is the winner of all comparisons with the other algorithms. We demonstrated that the performance of an OFDM system suffering from nonlinear distortions, caused by non-stationary HPA with memory, can be greatly improved by the proposed adaptive neural pre-distorter using Levenberg-Marquardt algorithm which proved to be efficient.

6. REFERENCES

[1] R. Zayani, R. Bouallegue and D. Roviras, “An Adaptive Neural Network Pre-Distorter for non stationary HPA in OFDM Systems”, 15th European Signal Processing Conference EUSIPCO, September 3-7 2007, Poznan, Poland.

[2] E. Castillo, B. Berdinas, O. Romero and A. Betanzos, “A Very Fast Learning Method for Neural Networks Based on Sensitivity Analysis”, Journal of Machine Learning Research 7 (2006) 1159–1182.

[3] Y. Ding, I. Nilkhamhang and A. Sano, “An Adaptive Nonlinearity Compensation Scheme for OFDM Communication Systems”, Proceedings of the 6th World Congress on Intelligent Control and Automation, June 21 - 23, 2006, Dalian, China.

[4] R. Zayani, R. Bouallegue, “A neural Network Pre-distorter for the compensation of HPA non-linearity”, International Journal of Coputer Science and Network Security, IJCSNS March 2007.

[5] Y. Wang, S. Kim, J. C. Principe, “Comparison of TDNN Training Algorithms in Brain Machine Interfaces”, International Joint Conference on Neural Networks , IJCNN 2005.

[6] Udo Seiffert, “Training of Large-Scale Feed-Forward Neural Networks”, International Joint Conference on Neural Networks, IJCNN 2006.

[7] A.A.M. Saleh, Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers, IEEE Trans. Commun. 29 (1981) 1715–1720.

[8] J. li, J. Ilow, “Adaptive Volterra predistorters for compensation of non-linear effects with memory in OFDM transmitters”, Proceeding of the 4th Annual Communication Networks and Services Research Conference (CNSR’06), 2006.