# APPLICATION OF THE BLOCK RECURSIVE LEAST SQUARES ALGORITHM TO ADAPTIVE NEURAL BEAMFORMING

E.D.Di Claudio, R.Parisi, and G.Orlandi
INFOCOM Department
University of Rome " La Sapienza" Via Eudossiana 18 - I-00184 Rome, Italy.
email: parisi@infocom.ing.uniroma1.it, dic@infocom.ing.uniroma1.it
Ph.: +39-6-44585837, Fax: +39-6-4873300

## ABSTRACT

**Spatial beamforming using a known training sequence is a well-understood technique for canceling uncorrelated interferences from telecommunication signals [1]. Most of on-line adaptive beamforming algorithms are based on linear algebra and linear signal models. Anyway both in the transmitter amplifier and in the array receiver nonlinearities may arise, producing distorted waveforms and reducing the performance of the demodulation process. A nonlinear spatial beamformer with sensor arrays may use a neural network to cope with communication system nonlinearities.**
**In this work we show that a feedforward neural network trained with a LS-based algorithm may get the convergence in a time suitable to most applications.**

## 1. INTRODUCTION

Signal processing by sensor arrays is sought as a technique for improving location parameter estimation (high resolution algorithms) and increasing the capacity of telecommunication links. The key feature of sensor arrays is that the number of processed sources and the Signal-to-Noise-Ratio (SNR) are limited in principle only by the system size [2].
In particular an array is able to create a spatial gain pattern with somewhat arbitrary shape by properly combining the outputs, according to a specified optimization criterion. Different patterns can be formed in parallel from the same received signals, enabling simultaneous demodulation and interference suppression in a multiple source environment.
If the beam computation is realizable by an on-line approach, the capacity of the communication system can be substantially improved. This requires adaptive algorithms able to converge to the steady-state solution before link parameters change. This is a serious problem in mobile communication systems, that may suffer also from the presence of nonlinearities along the signal path [3] and impulsive interferences, thus requiring a proper nonlinear treatment [4].
The signal model at the array output is represented by the classical linear equation [2]:

0-7803-4256-9/97/$10.00 ©1997 IEEE 

$$\mathbf{x}(t) = \mathbf{A}\begin{bmatrix} \mathbf{s}(t) \\ \mathbf{i}(t) \end{bmatrix} + \mathbf{n}(t) \tag{1}$$

where **x**(t) is the M-by-1 sensor snapshot vector at time t, **s**(t) is the ideal K-by-1 (K<M) signal vector, **A** represents the (unknown) M-by-Q array steering matrix [2], **i**(t) is the (Q-K)-by-1 interference (K≤Q<M) and **n**(t) is the additive background noise, uncorrelated with source and interference. All involved signals are assumed to be complex-valued (analytic signals) [1],[2].
Adaptive spatial beamforming often uses a preamble training sequence **y**(t) to recognize useful signals and cancel interferences by steering array nulls toward disturbances [1].
**y**(t) may be an analog local replica of the useful signal(s) **s**(t), or the sequence of modulated symbol vectors $\{\mathbf{a}_i, i=1,2,..,N\}$ which forms **s**(t) [3]:

$$s(t) = \sum_{i=1}^{N} \mathbf{a}_i \mathbf{u}(t - iT) \tag{2}$$

The goal is to choose the proper parameters of a function $F(\mathbf{x}(t))$ of the array output which best approximates **y**(t). In classical beamforming the function is linear and is expressed as the Hermitian product with a weight vector **w**:

$$\mathbf{y}(t) = \mathbf{w}^H \mathbf{x}(t) + \mathbf{e}(t), \tag{3}$$

where **e**(t) is the approximation error.
Due to channel non-stationarity and communication efficiency requirements the adaptation process of **w** should be fast. The most widely known algorithms for on-line adaptation are based on the stochastic gradient descent, or Least Mean Squares (LMS) [5]. Anyway, the rate of convergence of LMS is linear [6] and is bounded by $(\rho-1)^2/(\rho+1)^2$, where $\rho$ is the condition number of the Hessian matrix [6].
As a matter of fact, the Hessian matrix in narrowband adaptive beamforming coincides with the (scaled) spatial cross sensor correlation matrix (CSCM) of sensor outputs. Its condition number is of the same order of the array SNR and can be very high ($10^2 \div 10^9$) in telecommunication applications. For this reason linear beamformers frequently use methods based on Recursive Least Squares (RLS) to get higher rates of convergence with respect to classical gradient-based approaches [5].

## 2. NEURAL BEAMFORMING

In communication systems the adaptive array is part of a chain of blocks; most of them are intrinsically nonlinear or may exhibit undesired nonlinearities (amplifiers, mixers, clampers, ...). In order to cope with these nonlinearities, a

nonlinear beamformer should be employed. Feedforward multilayer neural networks may provide a solution to this problem [7] in a straightforward approach.
The input-output relationship of the proposed neural network beamformer with M sensors is:

$$\mathbf{y}(t) = F\{\mathbf{x}(t)\} + \mathbf{e}(t) \tag{4}$$

In this case the correct model for **x**(t) is related to **s**(t) by the equation:

$$\mathbf{x}(t) = \mathbf{A}G\left\{\begin{bmatrix}\mathbf{s}(t)\\ \mathbf{i}(t)\end{bmatrix}\right\} + \mathbf{n}(t) \tag{5}$$

In this formula G{.} is an unknown nonlinear M-by-Q matrix transfer function, that should be inverted by the neural beamformer.
Using the $L_2$ norm, the error functional $J$ to be minimized is:

$$J = E\{trace[\mathbf{e}(t)\mathbf{e}^{\mathrm{H}}(t)]\} \tag{6}$$

where $E\{.\}$ denotes the expectation operator over time, *trace*[.] is the matrix trace operator and $(.)^{\mathrm{H}}$ indicates Hermitian transposition.
The neural beamformer realizes a nonlinear memoryless functional of **x**(t). A standard multilayer perceptron (MLP) can approximate arbitrary input-output relationships of this kind [7]. The input of the MLP are the sensor outputs, while **y**(t) contains the target signals.
The minimization of $J$ may be accomplished by separating the real and the imaginary parts of all signals and using a standard backpropagation (BP) approach [7]. As an alternative, the complex neuron model can be used, which gives some benefits for the reduced number of free weights [5],[7].
Backpropagation is a stochastic gradient descent method and is characterized by the same limitations of LMS [7]. Faster second-order [6] convergence can be obtained with the Block Recursive Least Squares (BRLS) approach described in [8],[9]. In [9] BRLS is shown to be a Newton-type algorithm able to reach convergence with a very favorable numerical conditioning.
In this work we show by numerical simulations how a neural beamformer trained with the BRLS technique can be very effective in the presence of high levels of noise and interference, while detecting and recovering multiple signals of interest.

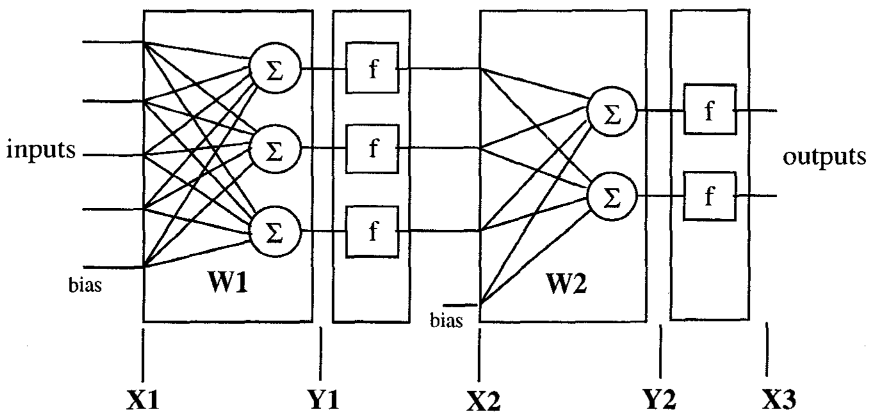


**Figure 1**: two-layer MLP.

## 3. SUMMARY OF THE BRLS ALGORITHM

The BRLS algorithm is an iterative learning procedure which solves an overdetermined linear system of equations for each layer of the network. With respect to similar algorithms, it minimizes the error functional at the linear summation nodes of the neurons (*descent in the neuron space* [9]), in contrast to the traditional optimization in the *weight space*.
At the n-th iteration and for the k-th layer we introduce the following matrices:

$$\mathbf{X}_k(n) = \begin{pmatrix} \mathbf{x}_{k,1}^T(n) \\ \dots \\ \mathbf{x}_{k,P}^T(n) \end{pmatrix} \mathbf{Y}_k(n) = \begin{pmatrix} \mathbf{y}_{k,1}^T(n) \\ \dots \\ \mathbf{y}_{k,P}^T(n) \end{pmatrix} \tag{7}$$

where $\mathbf{x}^T{}_{k,p}(n)$ and $\mathbf{y}^T{}_{k,p}(n)$ are the input and output row vectors of the linear section of the MLP layer, in the presence of the p-th learning pattern (P is the length of the whole batch). The lengths of $\mathbf{x}^T{}_{k,p}(n)$ and $\mathbf{y}^T{}_{k,p}(n)$ are respectively $(N_k+1)$ and $N_{k+1}$, being $N_k$ the number of input units to the layer; $\mathbf{X}_1(n)$ contains the external inputs, while $\mathbf{X}_{L+1}(n)$ contains the global outputs of the net (see fig. 1 for the case L=2).
Matrices $\mathbf{Y}_k$ are computed in a forward-propagation step through the k-th layer, similar to that of standard BP [7]:

$$\mathbf{X}_k(n)\mathbf{W}_k(n) = \mathbf{Y}_k(n) \tag{8}$$

while the passage through the nonlinearities is represented by the following:

$$\mathbf{X}_{k+1}(n) = \left[f(\mathbf{Y}_k(n)) \quad \mathbf{1}\right] \tag{9}$$

f(.) represents the activation function and **1** is the column of the bias inputs. At the first iteration weight matrices $\mathbf{W}_k$ are initialized at random.
The optimization in the neuron space is performed separately for each layer following the formula:

$$\hat{\mathbf{Y}}_k(n) = \mathbf{Y}_k(n) + \mathbf{D}_k(n) \tag{10}$$

where $\hat{\mathbf{Y}}_k$ is an estimation of the desired $\mathbf{Y}_k$ based on the *direction matrix* $\mathbf{D}_k$.
$\mathbf{D}_k$ may be chosen in several ways, depending on the method being adopted. The simplest choice is the negative of the *gradient matrix*:

$$\hat{\mathbf{Y}}_k(n) = \mathbf{Y}_k(n) - \eta \nabla_{\mathbf{Y}_k} E \tag{11}$$

where $\eta$ is a proper *correction factor*. The derivatives of $E$ w.r.t. the **y**'s are computed using formulas similar to those of BP [7].
Given the estimate $\hat{\mathbf{Y}}_k$, the new weight matrix for each layer is computed from the Least Squares (LS) solution of the following system:

$$\mathbf{X}_k(n)\mathbf{W}_k(n+1) = \hat{\mathbf{Y}}_k(n) \tag{12}$$

where in particular QR or SVD based algorithms can be used [5]. Formula (12) represents the general formulation of the class of LS-based learning algorithms; it consists in perturbing the matrix $\mathbf{Y}_k$ in order to recover the consistency of the system in the LS sense. This gives the new weight matrix $\mathbf{W}_k(n+1)$, to be used in the next forward-propagation step.
In order to stabilize learning in earlier steps, when weight are far from optimal values, a recursive implementation can be adopted, updating the solution by the classical on-line RLS-QR algorithm [5],[9]. In this case the forward and backpropagation phases act on the last block of snapshots [2]. A proper exponential forgetting factor $\lambda$ can be used to discard the influence of older samples [5]. More details about the BRLS algorithm can be found in [9]. Here we point out that the numerical robustness of the algorithm is threatened by the severe ill-conditioning of matrix $\mathbf{X}_1$ which is just the square root of the array CSCM. However the use of a square root formulation keeps the condition number acceptably low ($10^4 \div 10^5$), allowing the use of the limited precision floating-point arithmetic offered by commercial DSP microprocessors.

## 4. EXPERIMENTAL TRIALS

In the proposed computer experiment a linear equispaced array of ten sensors, with an intersensor spacing of half wavelength, is used as receiver. The useful signals are two independent 8-QAM waveforms with a SNR of 10 dB w.r.t. each sensor, impinging from 8 and 25 degrees of azimuth, referred to the array broadside; the elevation is zero degrees for both sources. The signals of interest are distorted by a memoryless arctangent nonlinearity, which models the amplifier static transfer function. The interference is represented by a white Gaussian noise source, coming from a direction of 15 degrees, with a SNR of 20 dB. The array receiver gains fluctuate with a standard deviation of 1% of their nominal values during the experiment. The background noise is supposed to be Gaussian, white and isotropic [2]. The conditions of the experiment are recognized to be rather unfavorable since all coherent sources are within one array beamwidth [2], [3].

The neural network used in the experiment is a multilayer perceptron with 20 inputs, 8 hidden neurons and four outputs, with sigmoidal-type nonlinearities. Learning was performed with the BRLS algorithm on 100 epochs of 30 snapshots each. Several values for the forgetting factor $\lambda$ were tried; in the described experiment $\lambda$=0.99 was used. The following figure shows the curve of the ratio of the target signal power (Mean Squared Signal, MSS) to the error power (Mean Squared Error, MSE) for each source of interest during the learning. The steady state solution is reached after about 35 epochs.

The almost monotonic shape of the learning curves demonstrates the ability of the BRLS algorithm to deal with ill-conditioned problems and to track the short time channel fluctuations that can be expected in real systems. Also remarkable is the insensitivity of the BRLS method to the starting guess of the network weights [9], which is essential for successful signal processing applications.

## CONCLUSION

The recently introduced BRLS algorithm for fast training MLP networks allows the use of neural architectures in challenging multichannel DSP problems, characterized by severe ill-conditioning of the data matrix coupled with stringent requirements on convergence rate. The general approach described in [9] has a great flexibility in changes of the error functional and learning parameters, and may introduce several forms of weight regularization through system (12) [5]. We plan to apply the BRLS neural approach to combine space-time equalization of communication channels and nonlinear distortion correction.

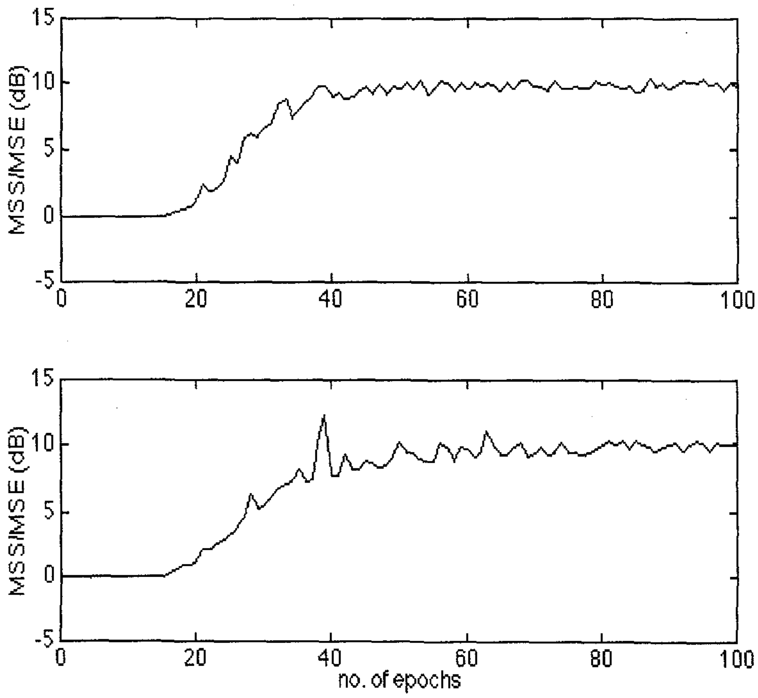


**Figure 2:** Learning curves for the two complex outputs.

## REFERENCES


[1] D.H.Johnson and D.E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, Prentice-Hall, 1993.

[2] R.O.Schmidt, "Multiple Emitter Location and Signal Parameter Estimation," *IEEE Trans. on Antennas and Propagation*, Vol. 34, No. 3, pp.276-280, March 1986.

[3] E.A.Lee, D.G.Messerchmitt, *Digital Communication*, Kluwer Academic Publishers, Boston, USA, 1988.

[4] L.P.Ammann, ``Statistically robust signal subspace identification," *Proc. of the Int. Conf. on ASSP*, ICASSP'90, pp 2711-2714, 1991.

[5] S.Haykin, *Adaptive Filter Theory,* Prentice Hall, 2nd ed. 1991.

[6] D.G.Luenberger, *Linear and Nonlinear Programming*, Addison Wesley,2nd ed. 1989.

[7] S.Haykin, *Neural Networks-A Comprehensive Foundation,* IEEE Press, 1994.

[8] E.D.Di Claudio, R. Parisi, G. Orlandi, "Block-recursive least square technique for training multilayer perceptrons," *1994 World Conference on Neural Networks (WCNN'94)*, San Diego, May 26-29, 1994.

[9] R.Parisi, E.D. Di Claudio, G. Orlandi and B.D. Rao, "A generalized learning paradigm exploiting the structure of feedforward neural networks," *IEEE Trans. on Neural Networks*, vol.7, no.6, November 1996.