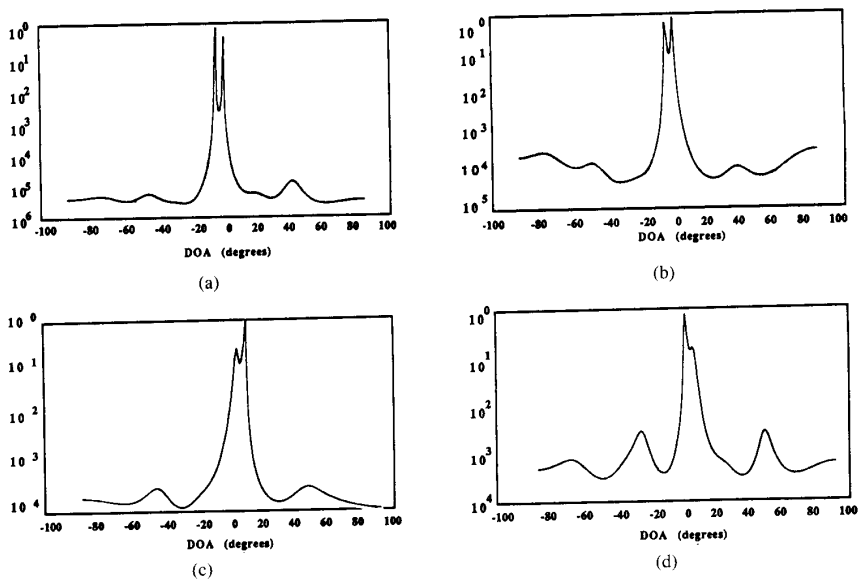

Fig. 2. Noise space ITM performance versus SNR. (a) Noise space ITM (30-dB SNR—3 iterations). (b) Noise space ITM (20-dB SNR—3 iterations). (c) Noise space ITM (10-dB SNR—3 iterations. (d) Noise space ITM (0-dB SNR—3 iterations).

This function has three fixed points

$$\left(0, \frac{1 - \sqrt{1 - 4a^2}}{2}, \frac{1 + \sqrt{1 - 4a^2}}{2}\right).$$

The values 0 and $1 + \sqrt{1 - 4a^2}/2$ are stable fixed points whereas $1 - \sqrt{1 - 4a^2}/2$ is unstable. Therefore, if $x_0 < 1 - \sqrt{1 - 4a^2}/2$ then $x_n \to 0$ or $1/x_n \to \infty$ and if $x_0 > 1 - \sqrt{1 - 4a^2}/2$ then $x_n \to 1 + \sqrt{1 - 4a^2}/2$ or $1/x_n \to 2/1 + \sqrt{1 + 4a^2}$. The statement of the Lemma follows.

Proof of Lemma 2: The same as Lemma 1 with the exception that $x_0 = 1/\lambda$.

References

[1] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propogat.*, vol. AP-34, pp. 276–280, 1986.

[2] R. Schmidt, "A signal subspace approach to multiple source location and spectral estimation," Ph.D. dissertation, Stanford University, Stanford, CA, 1981.

[3] G. Bienvenue, "Eigensystem properties of the sample space correlation matrix," in *Proc. IEEE ICASSP'83* (Boston, MA), pp. 81–84.

[4] M. Wax, T.-J. Shan, and T. Kailath, "Location and spectral density estimation of multiple sources," in *Proc. 16th Asilomar Conf. Circuits Syst. Comput.* (Pacific Grove, CA), pp. 322–326.

[5] A. Di, "Multiple source location—A matrix decomposition approach," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, no. 4, pp. 1086–1091, 1985.

[6] D. H. Johnson and S. R. DeGraff, "Improving the resolution of bearing in passive sonar arrays by eigenvalue analysis," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, no. 4, pp. 638–647, 1982.

[7] R. Kumerasan and D. W. Tufts, "Estimating the angles of arrival of multiple plane waves," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-19, pp. 134–139, 1983.

[8] H. Wang and M. Kaveh, "On the performance characterization of signal-subspace processing," in *Conf. Rec. 19th Asilomar Conf. Circuits, Syst. Comput.* (Pacific Grove, CA), Nov. 6–8, 1985. Washington, DC: IEEE Computer Society Press, 1986.

The Complex Backpropagation Algorithm

Henry Leung and Simon Haykin

***Abstract*—The backpropagation algorithm provides a popular method for the design of multilayer neural networks. This correspondence generalizes the backpropagation algorithm to include complex coefficients and complex signals.**

I. Introduction

Adaptive signal processing is a fast growing area in signal processing. Its applications include equalization, beamforming, noise cancellation, and detection. The advantage of using adaptive techniques is to make the system capable of handling a changing environment. Many powerful adaptive algorithms have been developed such as the least mean squares (LMS) algorithm, the recursive least squares (RLS) algorithm, and the recursive QR-decomposition least squares (QRD) algorithm. Each of these algorithms has its own advantages and disadvantages: some of them are fast convergent but numerically unstable, or computationally simple but slow convergent. However, all these conventional adaptive signal processing algorithms have a major handicap, that is, the underlying assumption of linearity.

Conventional adaptive filtering algorithms are all derived from a model assuming that the output is a linear function of the tap inputs. This assumption is a simple and general approximation to a particular problem, and usually yields a reasonable result. However, we may obtain a better performance by introducing certain

Manuscript received November 15, 1989; revised March 3, 1991.

The authors are with the Communications Research Laboratory, McMaster University, Hamilton, Canada L8S 4K1.

IEEE Log Number 9101318.

modifications. For instance, consider the problem of adaptive detection of a target in sea clutter. The use of a linear adaptive filter may not be successful since it is well known that sea clutter has a K-distribution, which implies that it cannot be generated by a linear process. Other similar situations exist in signal processing such as the adaptive equalization of a fading channel, non-Gaussian noise cancellation, and so on. Thus, nonlinear adaptive signal processing becomes a challenging problem.

Recently the rediscovery of the backpropagation algorithm [1] has drawn a great deal of attention because of its universal approximation ability. It has been successfully applied in neural networks for classification, and prediction. We therefore would like to apply this nonlinear adaptive algorithm to the signal processing problems, that require a complex mathematical representation of the signal. For instance, the data received by a coherent radar has two components; namely, amplitude and phase. Thus, it is the purpose of this article to generalize the backpropagation algorithm to include complex signal and coefficients, so that we can apply it to general radar signal processing and communications problems.

II. Theory

The backpropagation (BP) algorithm is one of the most popular learning algorithms in neural networks. This algorithm performs an approximation to the global minimization achieved by the method of steepest descent. As such, the algorithm may be viewed as a generalization of the least mean squares (LMS) adaptive algorithm. The main difference between BP and LMS is the underlying model; namely, LMS occupies an adaptiver linear combiner, whereas BP works on a multilayer perceptron (Fig. 1).

A multilayer perceptron consists of many adaptive linear combiners, each of which has a nonlinearity at its output as shown in Fig. 2. The input/output relationship of such a unit is characterized by the nonlinear recursive difference equation

$$x_i^{(l+1)} = f\left(\sum_{j=1}^{N} w_{ij}^{(l)} x_j^{(l)} + \theta_i^{(l)}\right) \tag{1}$$

and this relation is generalized to all units in multilayer perceptron as listed in Fig. 1.

The error signal ϵ_j required for adaptation is defined as the difference between the desired response and the output of the perceptron:

$$\epsilon_j(n) = d_j(n) - y_j(n) \qquad j = 1, 2, \cdots, N_M \tag{2}$$

where $d_j(n)$ is the desired response at the jth node of the output layer at time n; $y_j(n)$ is the output at the jth node of the output layer, and N_M is the number of nodes in the Mth layer or the output layer. Hence, the sum of error squares produced by the network is

$$E(n) = \sum_{j=1}^{N_M} \epsilon_j(n)\epsilon_j^*(n). \tag{3}$$

The BP algorithm minimizes the cost functional $E(n)$ by recursively altering the coefficient $\{w_{ij}^{(l)}, \theta_i^{(l)}\}$ based on the gradient search technique. Thus, finding the gradient vector of $E(n)$ is the main idea of deriving the BP algorithm. We first find the partial derivative of $E(n)$ with respect to the coefficients of the output layer, and then extend to the coefficient of all hidden units. Since $E(n)$ is a real-valued function which is not analytic, we need to derive the partial derivative of $E(n)$ with respect to the real and imaginary part of the coefficients separately. Writing $w_{ij}^{(l)}(n)$ as

$$w_{ij}^{(l)}(n) = wr_{ij}^{(l)}(n) + iwi_{ij}^{(l)}(n) \tag{4}$$

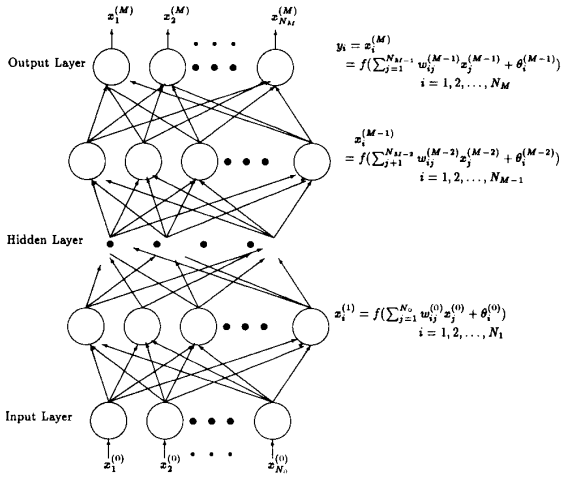

Fig. 1. Multilayer perceptron.

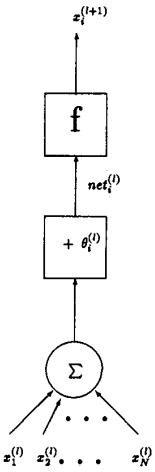

Fig. 2. Adaptive nonlinear combiner.

our purpose is to obtain $\partial E(n)/\partial wr_{ij}^{(l)}$ and $\partial E(n)/\partial wi_{ij}^{(l)}$. First, let us consider the adaptation rule of the output layer

$$wr_{ij}^{(M-1)}(n+1) = wr_{ij}^{(M-1)}(n) - \frac{1}{2}\mu \frac{\partial E(n)}{\partial wr_{ij}^{(M-1)}(n)} \tag{5}$$

$$wi_{ij}^{(M-1)}(n+1) = wi_{ij}^{(M-1)}(n) - \frac{1}{2}\mu \frac{\partial E(n)}{\partial wi_{ij}^{(M-1)}(n)}. \tag{6}$$

Combining (5) and (6), we have

$$wr_{ij}^{(M-1)}(n+1) + iwi_{ij}^{(M-1)}(n+1) = wr_{ij}^{(M-1)}(n) + iwi_{ij}^{(M-1)}(n) - \frac{1}{2}\mu\left(\frac{\partial E(n)}{\partial wr_{ij}^{(M-1)}(n)} + i\frac{\partial E(n)}{\partial wi_{ij}^{(M-1)}(n)}\right) \tag{7}$$

or

$$w_{ij}^{(M-1)}(n+1) = w_{ij}^{(M-1)}(n) - \frac{1}{2}\mu\left(\frac{\partial E(n)}{\partial wr_{ij}^{(M-1)}(n)} + i\frac{\partial E(n)}{\partial wi_{ij}^{(M-1)}(n)}\right) \tag{8}$$

Thus, the next step is to find some expressions for the partial derivative:

$$\frac{\partial E(n)}{\partial wr_{ij}^{(M-1)}} = \frac{\partial E(n)}{\partial y_i} \cdot \frac{\partial y_i}{\partial net_i^{(M)}} \cdot \frac{\partial net_i^{(M)}}{\partial wr_{ij}^{(M-1)}} + \frac{\partial E(n)}{\partial y_i^*} \cdot \frac{\partial y_i^*}{\partial net_i^{(M)*}} \cdot \frac{\partial net_i^{(M)*}}{\partial wr_{ij}^{(M-1)}} \tag{9}$$

where

$$y_i = f(net_i^{(M)}) \tag{10}$$

$$net_i^{(M)} = \sum_{j=1}^{N_{M-1}} w_{ij}^{(M-1)} x_j^{(M-1)} + \theta_i^{(M-1)} \tag{11}$$

and f can be any nonlinear function in principle. We take f as the sigmoidal function $1/1 + ext(-net_i)$, which is the most common one used in multilayer network. The exponential function is now used in its complex version, and the principal values are used. One point to stress is that if we use the exponential function, we need to take care of the problem of singularities. We avoid this problem by scaling the input data to some region in the complex plane.

One point that should be noted is the absence of the partial derivatives $\partial y_i/\partial net_i^*$ and $\partial y_i^*/\partial net_i$ in (9). More precisely, there should be two more terms in (9) when the chain rule is applied. When we look at the definition of y_i, the function f is a sigmoidal function which does not contain any complex operation. That is, f maps a real number to a real number and a complex number to a complex number, but it will not map a real number to a complex number. Thus

$$x_i^{*(l+1)} = f(net_i^{*(l+1)}) \qquad l = 0, 1, \cdots, M-1 \tag{12}$$

and hence those two derivatives are equal to zero.

Evaluating all the partial derivatives in (9), we get

$$\frac{\partial E(n)}{\partial wr_{ij}^{(M-1)}} = -(d_i^* - y_i^*) f'(net_i) x_j^{(M-1)} - (d_i - y_i) f'(net_i^*) x_j^{*(M-1)}. \tag{13}$$

Similarly

$$\frac{\partial E(n)}{\partial wi_{ij}^{(M-1)}} = -i(d_i^* - y_i^*) f'(net_i) x_j^{(M-1)} + i(d_i - y_i) f'(net_i^*) x_j^{*(M-1)}. \tag{14}$$

Combining (13) and (14), we have

$$\frac{\partial E(n)}{\partial wr_{ij}^{(M-1)}} + i\frac{\partial E(n)}{\partial wi_{ij}^{(M-1)}} = -2(d_i - y_i) f'(net_i^*) x_j^{*(M-1)}. \tag{15}$$

Substituting (15) into (8), we obtain the adaptation rule for the output layer:

$$w_{ij}^{(M-1)}(n+1) = w_{ij}^{(M-1)}(n) + \mu(d_i(n) - y_i(n)) f'(net_i^*(n)) x_j^{*(M-1)}(n) \qquad i = j, \cdots, N_M; \quad j = 1, 2, \cdots, N_{M-1} \tag{16}$$

where N_M and N_{M-1} are the numbers of nodes in layer M (output layer) and layer $M-1$ (the hidden layer converted to the output layer), respectively. We now apply the same procedure to the $(M-1)$th layer. The generalization to other hidden layers is trivial and will be given at the end of this article. The adaptation rule for this hidden layer is the same as (8). Thus, what we need to do is to find the partial derivatives

$$\frac{\partial E(n)}{\partial wr_{ij}^{(M-2)}} = \frac{\partial E(n)}{\partial x_i^{(M-1)}} \cdot \frac{\partial x_i^{(M-1)}}{\partial net_i^{(M-1)}} \cdot \frac{\partial net_i^{(m-1)}}{\partial wr_{ij}^{(M-2)}} + \frac{\partial E(n)}{\partial x_i^{*(M-1)}} \cdot \frac{\partial x_i^{*(M-1)}}{\partial net_i^{*(M-1)}} \cdot \frac{\partial net_i^{*(M-1)}}{\partial wr_{ij}^{(M-2)}}. \tag{17}$$

Since $E(n)$ is not related to $x_i^{(M-1)}$ or $x_i^{*(M-1)}$ explicitly, the evaluation of the above equation requires the use of the chain rule again. Consider

$$\frac{\partial E(n)}{\partial x_i^{(M-1)}} = -\sum_k (d_k^* - y_k^*) \frac{\partial y_k}{\partial x_i^{(M-1)}} \tag{18}$$

where

$$y_k = f(net_k^{(M)}) = f\left(\sum_{l=1}^{N_{M-1}} w_{kl}^{(M-1)} x_i^{(M-1)} + \theta_k^{(M-1)}\right). \tag{19}$$

Hence

$$\frac{\partial E(n)}{\partial x_i^{(M-1)}} = -\sum_k (d_k^* - y_k^*) f'(net_k^{(M)}) w_{ki}^{(M-1)}. \tag{20}$$

Similarly,

$$\frac{\partial E(n)}{\partial x_i^{*(M-1)}} = -\sum_k (d_k - y_k) f'(net_k^{*(M)}) w_{ki}^{*(M-1)}. \tag{21}$$

Substituting (20) and (21) into (17) we have

$$\frac{\partial E(n)}{\partial wr_{ij}^{(M-2)}} = \left[-\sum_k (d_k^* - y_k^*) f'(net_k^{(M)}) w_{ki}^{(M-1)}\right] \cdot f'(net_i^{(M-1)}) x_j^{(M-2)} + \left[-\sum_k (d_k - y_k) f'(net_k^{*(M)}) w_{ki}^{*(M-1)}\right] \cdot f'(net_i^{*(M-1)}) x_j^{*(M-2)}. \tag{22}$$

The partial derivative with respect to the imaginary part is then

$$\frac{\partial E(n)}{\partial wi_{ij}^{(M-2)}} = i\left[-\sum_k (d_k^* - y_k^*) f'(net_k^{(M)}) w_{ki}^{(M-1)}\right] f'(net_i^{(M-1)}) x_j^{(M-2)} - i\left[-\sum_k (d_k - y_k) f'(net_k^{*(M)}) w_{ki}^{*(M-1)}\right] \cdot f'(net_i^{*(M-1)}) x_j^{*(M-2)}. \tag{23}$$

Combining (22) and (23) into complex form yields the adaptation rule

$$w_{ij}^{(M-2)}(n+1) = w_{ij}^{(M-2)}(n) + \mu\left[\sum_k (d_k - y_k) f'(net_k^{*(M)}) w_{ki}^{*(M-1)}\right] \cdot f'(net_i^{*(M-1)}) x_j^{*(M-2)}. \tag{24}$$

Equations (24) and (16) are the two basic update equations for the BP algorithm. Since the logic of deriving the update formula for other hidden units is exactly the same, we will end this derivation by just listing the complex BP algorithm in Table I, as the complex version of that given in [2].

To see whether the above complex backpropagation algorithm works or not, we consider a simple classification problem. In [1], there is an application of the backpropagation algorithm to classify a 4-2-4 problem. {1 0 0 0}, {0 1 0 0}, {0 0 1 0}, {0 0 0 1} are submitted to a three layer network with four input nodes, two hidden nodes, and four output nodes to see how well the network can classify these patterns. We follow this idea and input complex sequences, that is, {(1, 1) (0, 0) (0, 0) (0, 0)}, {(0, 0), (1, 1) (0, 0)

TABLE I
THE COMPLEX BACKPROPAGATION TRAINING ALGORITHM

1) Initialize weights and offsets.
Set all weights and node offsets to small complex random values.
2) Present input and desired outputs.
Present input vector $\vec{x}(0), \vec{x}(1), \cdots, \vec{x}(n)$ and desired response $d(0), d(1), \cdots, d(n)$ where n is the number of training patterns.
3) Calculate actual outputs.
Using the formula in Fig. 1 to calculate outputs $y_1, y_2 \cdots, y_{N_M}$.
4) Adapt weights:

$$w_{ij}(t+1) = w_{ij}(t) + \mu\delta_i x_j^*$$

where x_j' = output of node j or input to node i

$$\delta_i = \begin{cases} (d_i - y_i) f^{*\prime} & \text{output layer} \\ f^{*\prime} \sum_k w_{ik}^* & \text{input layer.} \end{cases}$$

Note that the momentum term can also be added to the above formula.

TABLE II
CLASSIFICATION USING THE COMPLEX BACKPROPAGATION ALGORITHM

Exact	200 Trainings	2000 Trainings	3000 Trainings
(1, 1)	(0.519173, 0.486552)	(1.001064, 0.997023)	(1.000026, 0.999944)
(0, 0)	(0.510598, 0.489922)	(2.38E − 03, 6.26E − 07)	(3.98E − 05, −6.35E − 06)
(0, 0)	(9.06E − 04, 0.00E + 00)	(−3.21E − 04, 0.00E + 00)	(−4.61E − 06, 0.00E + 00)
(0, 0)	(−2.67E − 02, 0.00E + 00)	(−4.68E − 03, 0.00E + 00)	(−7.53E − 05, .0.00E + 00)
(0, 0)	(0.464267, 0.495845)	(−8.22E − 04, 2.31E − 03)	(−1.79E − 05, 3.91E − 05)
(1, 1)	(0.506931, 0.526976)	(0.998151, 0.9999998)	(0.999969, 1.000001)
(0, 0)	(−8.57E − 04, 0.00E + 00)	(2.49E − 04, 0.00E + 00)	(4.41E − 06, 0.00E + 00)
(0, 0)	(2.59E − 02, 0.00E + 00)	(3.15E − 03, 0.00E + 00)	(5.83E − 05, 0.00E + 00)

(0, 0). We use two patterns rather than four simply because it should be sufficient for the purpose of this article. The result is depicted in Table II. We can see that the network can indeed classify complex signals. The generalization of the BP to deal with complex signals should make it possible to expand the line of applications of this powerful nonlinear signal processing algorithm.

REFERENCES

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, D. E. Rumelhart and J. L. McCelland, Eds. Cambridge, MA: Foundations M.I.T. Press, 1986.
[2] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, pp. 4–22, Apr. 1987.

Optimum Laguerre Networks for a Class of Discrete-Time Systems

Mohammad A. Masnadi-Shirazi and N. Ahmed

***Abstract*—We present an analytical solution to the problem of designing optimum Laguerre networks for the class of discrete-time systems whose unit-sample response is of the form $\tilde{h}(n) = \sum_{i=1}^{p} A_i b_i^n$, where $A_i > 0$ and $0 < b_i < 1$ are real numbers. This is achieved by minimizing the mean-square error between the unit-sample response of the given discrete system and that of its Laguerre counterpart.**

Manuscript received April 27, 1989; revised July 5, 1990. Parts of this paper were presented at the IEEE International Conference on Systems.

M. A. Masnadi-Shirazi is with the Marine Physical Laboratory, Scripps Institution of Oceanography, University of California, San Diego, La Jolla, CA 92093-0205.
N. Ahmed is with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131.

IEEE Log Number 9101328.

I. INTRODUCTION

Synthesis of a linear time-invariant system by expanding its impulse response in terms of the orthonormal functions is an important application of such functions in the theory of linear systems [1]. We restrict our attention to discrete-time (DT) systems that are causal, linear, and time invariant. The input–output relation of such systems is given by the convolution sum

$$y(n) = \sum_{m=0}^{\infty} \tilde{h}(m) x(n-m) \tag{1}$$

where n is the time index, $x(n)$ and $y(n)$ are the input and output sequences, respectively, and $\tilde{h}(n)$ is the unit-sample response.

Let us now consider the case where the unit-sample response $\tilde{h}(n)$ is expanded in terms of a set of orthonormal sequences $\{\phi_k(n)\}$, to obtain

$$\tilde{h}(n) = \sum_{k=0}^{\infty} \tilde{c}_k \phi_k(n) \tag{2a}$$

where

$$\tilde{c}_k = \sum_{n=0}^{\infty} \tilde{h}(n) \phi_k(n) \tag{2b}$$

is the kth coefficient of the expansion. Substitution of (2a) in (1) and an interchange of the order of summation leads to

$$y(n) = \sum_{k=0}^{\infty} \tilde{c}_k v_k(n) \tag{3a}$$