A Fully Adaptive Normalized Nonlinear Gradient Descent Algorithm for Complex-Valued Nonlinear Adaptive Filters

Andrew Ian Hanna and Danilo P. Mandic, *Member, IEEE*

***Abstract*—A fully adaptive normalized nonlinear complex-valued gradient descent (FANNCGD) learning algorithm for training nonlinear (neural) adaptive finite impulse response (FIR) filters is derived. First, a normalized nonlinear complex-valued gradient descent (NNCGD) algorithm is introduced. For rigour, the remainder of the Taylor series expansion of the instantaneous output error in the derivation of NNCGD is made adaptive at every discrete time instant using a gradient-based approach. This results in the fully adaptive normalized nonlinear complex-valued gradient descent learning algorithm that is suitable for nonlinear complex adaptive filtering with a general holomorphic activation function and is robust to the initial conditions. Convergence analysis of the proposed algorithm is provided both analytically and experimentally. Experimental results on the prediction of colored and nonlinear inputs show the FANNCGD outperforming other algorithms of this kind.**

***Index Terms*—Adaptive filtering, nonlinear complex-valued filtering, normalized gradient descent, prediction.**

I. Introduction

ADAPTIVE filtering techniques are an important facet to many scientific disciplines such as communications, biomedical engineering, and life sciences. As these areas developed so did the character class of processed data. The majority of these diverse data existed in the real domain; however, increasing amounts started to root in the complex domain. This in turn lead to the development of complex-valued learning algorithms for nonlinear adaptive filters. For linear complex adaptive filtering, the complex least mean square (CLMS) algorithm [1] was developed. As the architectures of nonlinear neural network models became more involved, the complex backpropagation (CBP) algorithm was derived [2]–[5]. The complication with the CBP algorithm is finding a suitable activation function that is analytic and completely bounded in the complex plane [6]. Liouville's theorem states *"a bounded entire function in the complex domain is a constant"* [6]–[8], and so, to be able to employ gradient descent-based algorithms, a fully complex activation function must be analytic and bounded almost everywhere in the complex domain $\mathbb{C}$ for which there are many choices. Originally, a split complex activation was used in the processing of complex-valued signals.[1] However a split complex activation function cannot be analytic. To this cause, it is illustrated in [6] that the class of transcendental functions can be used as fully complex-valued activation functions successfully. For practical purposes, a complex-valued activation function proposed in [3] is frequently used.

For nonlinear adaptive filtering applications, a simple extension of an FIR filter is a dynamical perceptron, which is in fact an FIR filter superseded with a continuous nonlinear activation function. In Control Theory, this is also known as a Wiener model [7], [9]. Here, we consider such a filter realized as a dynamical complex neuron, as shown in Fig. 1.

A recent result provides novel ways of how to normalize the backpropagation algorithm [10]; however, for a highly ill-conditioned input correlation matrix, close to zero input vectors and signals with long time correlation and large dynamical range, it is difficult to choose the parameters of the algorithm for each particular case. In this paper, we embark upon the previously derived normalized nonlinear gradient descent (NNGD) algorithm [11] for real-valued adaptive filtering and extend it to be compliant with signals in the field of complex numbers $\mathbb{C}$. The NNGD algorithm is a member of the class of fully adaptive normalized nonlinear gradient descent algorithms, which in the linear real-valued case have been developed in [12]–[15]. The derivation of the NNGD algorithm [7] performs a Taylor series expansion of the instantaneous output error, which is then truncated leaving the driving terms of the algorithm. This results in a suboptimal algorithm due to an approximation of the expansion.

The choice of activation function, however, has major influences on the performance of algorithms for nonlinear filters. Therefore, based on the real-valued normalized nonlinear gradient descent algorithm, we first derive a normalized nonlinear complex-valued gradient descent (NNCGD) for a general complex-valued activation function. For rigour, we make the constant term, which is included to balance the truncated Taylor series expansion in the derivation of the NNCGD algorithm, adaptive using a gradient-based approach that produces the fully adaptive normalized nonlinear complex-valued gradient descent (FANNCGD) algorithm derived for a general holomorphic activation function. Experiments on the prediction of complex-valued colored and nonlinear signals show that the proposed algorithm outperforms the previously derived algorithms of this kind.

Manuscript received October 17, 2001; revised March 5, 2003. The associate editor coordinating the review of this paper and approving it for publication was Dr. Inbar Fijalkow.

A. I. Hanna is with the School of Computing Sciences, University of East Anglia, Norwich, NR4 7TJ, Norfolk, U.K. (e-mail: aih@cmp.uea.ac.uk.).

D. P. Mandic is with the Communications and Signal Processing Group of the Department of Electrical and Electronic Engineering, Imperial College of Science, Technology, and Medicine, London, U.K. (e-mail: d.mandic@ic.ac.uk).

Digital Object Identifier 10.1109/TSP.2003.816878

[1]For a complex number $\zeta = \alpha + j\beta$, a split complex activation is of the form $\Phi(\zeta) = \Phi(\alpha) + j\Phi(\beta)$, whereas a fully complex activation function is of the form $\Phi(\zeta) = \Phi^r(\alpha, \beta) + j\Phi^i(\alpha, \beta)$.

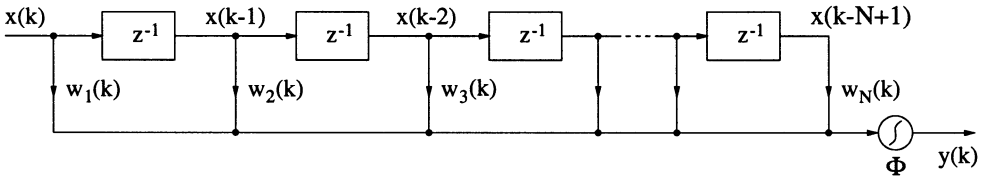


Fig. 1. Nonlinear FIR filter.

II. Normalized Nonlinear Complex Gradient Descent Algorithm

A. Nonlinear Complex Gradient Descent Algorithm

The equations that describe the nonlinear complex-valued gradient descent (NCGD) algorithm for a complex-valued dynamical perceptron, shown in Fig. 1, employed as a nonlinear FIR filter with a single output neuron are given by

$$e(k) = d(k) - y(k), \quad y(k) = \Phi(net(k)) \tag{1}$$

where $e(k)$ is the instantaneous output error of the filter at time instant k, $y(k)$ is the output from the complex-valued nonlinear activation function, $d(k)$ is the desired output, $\Phi(net(k))$ is some holomorphic function that is bounded almost everywhere in the complex domain [6], and

$$net(k) = \sum_{n=1}^{N} x_n(k) w_n(k) = \mathbf{x}^T(k)\mathbf{w}(k) \tag{2}$$

where $\mathbf{x}(k) \triangleq [x_1(k), \ldots, x_N(k)]^T$ denotes the complex input such that $x_n(k) = x(k-n+1)$, $n = 1, \ldots, N$ from Fig. 1. The complex weight vector is denoted by $\mathbf{w}(k) \triangleq [w_1(k), \ldots, w_N(k)]^T$, and N is the number of tap inputs. For simplicity, we state that

$$\Phi(net(k)) = \Phi^r(net(k)) + j\Phi^i(net(k)) = u(k) + jv(k) \tag{3}$$

where the superscripts $(\cdot)^r$ and $(\cdot)^i$, respectively, denote the real and imaginary parts of a complex quantity, and $j = \sqrt{-1}$. We can then split up the error term (1) into its real and imaginary parts as

$$e^r(k) = d^r(k) - u(k), \quad e^i(k) = d^i(k) - v(k) \tag{4}$$

$$E(k) = \frac{1}{2}|e(k)|^2 = \frac{1}{2}[e(k)e^*(k)] = \frac{1}{2}\left[(e^r)^2(k) + (e^i)^2(k)\right] \tag{5}$$

where $E(k)$ is the conventional cost function of the network [1], and $(\cdot)^*$ denotes the complex conjugate. The weight adaptation in the nonlinear complex gradient descent (NCGD) algorithm is therefore given by [3]

$$w_n(k+1) = w_n(k) + \Delta w_n(k), \quad n = 1, 2, \ldots, N \tag{6}$$

$$\begin{aligned} \Delta w_n(k) &= -\eta \nabla_w [E(k)]_{w=w_n(k)} \\ &= \eta e(k) \left(\Phi'[net(k)]\right)^* x_n^*(k) \end{aligned} \tag{7}$$

where η is the learning rate. The NCGD algorithm can be written in the compact form as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta e(k) \left(\Phi'[net(k)]\right)^* \mathbf{x}^*(k). \tag{8}$$

B. Normalized Nonlinear Complex Gradient Descent Algorithm

Input signals with unknown and possibly very large dynamical range, an ill-conditioned tap input autocorrelation matrix, and the coupling between different signal modes slow down the learning process. In order to speed up learning, it is desirable to calculate an optimal learning rate η that normalizes the model according to a minimization of the instantaneous output error at every iteration. The optimal learning rate of the NNCGD algorithm is calculated similarly to the real case [10], [11] by expanding the instantaneous output error by a Taylor series expansion

$$\begin{aligned} e(k+1) = e(k) &+ \sum_{n=1}^{N} \frac{\partial e(k)}{\partial w_n(k)} \Delta w_n(k) \\ &+ \sum_{n=1}^{N} \frac{\partial e(k)}{\partial x_n(k)} \Delta x_n(k) + \frac{\partial e(k)}{\partial d(k)} \Delta d(k) + \cdots \end{aligned} \tag{9}$$

The higher order terms of the polynomial can be neglected if $\Delta \mathbf{w}(k) = \|\mathbf{w}(k) - \mathbf{w}(k-1)\|$ is sufficiently small [16]; however, during the training period of the algorithm, this condition may not be held; thus, the term regarding higher order derivatives must be adjusted automatically. However, as an online learning model, we do not know the values of $x_n(k)$ and $d_n(k)$ *a priori*.[2] To this cause, we truncate the expansion to include the driving terms of the algorithm, namely, the weight vector, which gives

$$e(k+1) = e(k) + \sum_{n=1}^{N} \frac{\partial e(k)}{\partial w_n(k)} \Delta w_n(k) + h.o.t. \tag{10}$$

where $h.o.t.$ denotes the truncated terms of the expansion. Since $e(k)$ is a complex function, we can apply the Cauchy–Riemann equations to give[3]

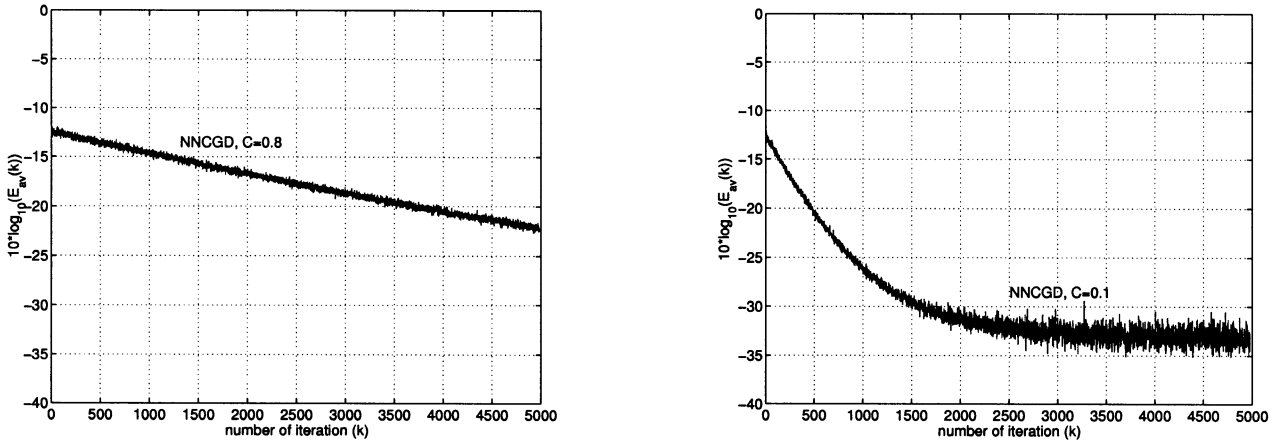
$$\frac{\partial e(k)}{\partial w_n(k)} = \frac{\partial e^i(k)}{\partial w_n^i(k)} - j\frac{\partial e^r(k)}{\partial w_n^i(k)} \tag{11}$$

and therefore

$$\frac{\partial e(k)}{\partial w_n(k)} = -\Phi'(net(k)) x_n(k). \tag{12}$$

[2]For instance, in unsupervised offline batch processing, the values of $d_n(k)$ are still unknown.

[3]For a full derivation, see Appendix A.



(a) (b)

Fig. 2. Convergence curves for NNCGD with varying C. (a) Convergence curve using NNCGD with $C = 0.1$. (b) Convergence curve using NNCGD with $C = 0.8$.

For simplicity, we take only the first two terms of (10), and substituting in (7) and (12) yields

$$\begin{aligned} e(k+1) \approx & e(k) - \sum_{n=1}^{N} x_n(k)\Phi'[net(k)]\eta x_n^*(k) \\ & \cdot (\Phi'[net(k)])^* e(k) \\ = & e(k)\Big[1 - \sum_{n=1}^{N} x_n(k)\Phi'[net(k)]\eta x_n^*(k) \\ & \cdot (\Phi'[net(k)])^*\Big] \\ = & e(k)\Big[1 - \eta \left|\Phi'[net(k)]\right|^2 \|\mathbf{x}(k)\|_2^2\Big]. \end{aligned} \tag{13}$$

For convenience, we employ the method given in [7] to solve for η. For the output error at time instant $(k+1)$ to be zero, the term in the square brackets must be zero, which gives the learning rate of the NNCGD algorithm as

$$\eta(k) = \frac{1}{\left|\Phi'(net(k))\right|^2 \|\mathbf{x}(k)\|_2^2 + C}. \tag{14}$$

In (14), C denotes a term added to balance the exclusion of second and higher order derivatives, which is denoted in (10) as $h.o.t.$ and the truncated terms in (9) from the Taylor series expansion. In the real-valued NNGD algorithm [11], this C term has been kept constant. The value of this term can have substantial effects on the convergence of the nonlinear adaptive filter, and the effects of this term will vary for different modes of application. To illustrate this, 500 independent simulations on the prediction of colored input were averaged to produce the convergence curves. The colored input was generated with complex-valued white noise $r(k)$ with zero mean and unit variance, which was then passed through a stable AR filter described by

$$\begin{aligned} a(k) = & 1.79a(k-1) - 1.85a(k-2) + 1.27a(k-3) \\ & -0.41a(k-4) + r(k). \end{aligned} \tag{15}$$

In each case, the order of the filter was $N = 10$, and the nonlinearity was the complex-valued hyperbolic tangent function, which was defined as

$$\Phi(x, \beta) = \frac{e^{\beta x} - e^{-\beta x}}{e^{\beta x} + e^{-\beta x}} \tag{16}$$

where $x \in \mathbb{C}$, and $\beta \in \mathbb{R}$. Fig. 2(a) shows the performance of the NNCGD algorithm with $C = 0.8$ reaching -22 dB, and Fig. 2(b) shows the NNCGD algorithm with $C = 0.1$ converging to -34 dB, which is an increase in performance of 12 dB, showing that the NNCGD algorithm is sensitive to the choice of C.

III. Fully Adaptive NNCGD Algorithm

The convergence curves in Fig. 2 clearly show a difference in performance according to varying values of C, which was added to balance the exclusion of the terms from (9). For this reason, it is proposed that an online adaptive term $C(k)$ from (14) be introduced, providing a fully adaptive normalized nonlinear complex gradient descent (FANNCGD) learning algorithm. The equation that defines the update of $C(k)$ is given by

$$C(k) = C(k-1) - \rho \nabla_C [E(k)]_{C=C(k-1)} \tag{17}$$

where ρ denotes the step size of the algorithm, and therefore

$$\begin{aligned} \nabla_C [E(k)]_{C=C(k-1)} = & \frac{\partial E(k)}{\partial C(k-1)} \\ = & \frac{1}{2}\Big[e(k)\frac{\partial e^*(k)}{\partial C(k-1)} + e^*(k)\frac{\partial e(k)}{\partial C(k-1)}\Big]. \end{aligned} \tag{18}$$

To calculate two partial derivative equations given in (18), it is necessary to use the Cauchy–Riemann equations to obtain[4]

$$\frac{\partial e^*(k)}{\partial C(k-1)} = -\Big[\frac{\partial \mathbf{w}(k)}{\partial C(k-1)}\Big]^* [\Phi'(net(k))]^* \mathbf{x}^*(k) \tag{19}$$

[4]For a full derivation of (19) and (20), see Appendix B.

and

$$\frac{\partial e(k)}{\partial C(k-1)} = -\frac{\partial \mathbf{w}(k)}{\partial C(k-1)}\Phi'(net(k))\mathbf{x}(k) \tag{20}$$

Writing the weight update term excluding the learning rate $\eta(k)$ to give

$$\Psi(k) = e(k)\left[\Phi'(net(k))\right]^* \mathbf{x}^*(k) \tag{21}$$

we can derive

$$\frac{\partial \mathbf{w}(k)}{\partial C(k-1)} = \frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)} + j\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)} \tag{22}$$

where we have (23) and (24), shown at the bottom of the page. Therefore

$$\begin{aligned}\frac{\partial \mathbf{w}(k)}{\partial C(k-1)} &= -\frac{\Psi^r(k-1)+j\Psi^i(k-1)}{\left[\left|\Phi'(net(k-1))\right|^2 \|\mathbf{x}(k-1)\|_2^2 + C(k-1)\right]^2} \\ &= -\frac{e(k-1)\left[\Phi'(net(k-1))\right]^* \mathbf{x}^*(k-1)}{\left[\left|\Phi'(net(k-1))\right|^2 \|\mathbf{x}(k-1)\|_2^2 + C(k-1)\right]^2}.\end{aligned} \tag{25}$$

For simplicity, we will denote $\partial \mathbf{w}(k)/\partial C(k-1) = -\xi(k)$. The gradient of the cost function with respect to the term $C(k)$ from (14) added to compensate for the truncation in (10) becomes

$$\begin{aligned}\frac{\partial E(k)}{\partial C(k-1)} &= \frac{1}{2}\left[e(k)\frac{\partial e^*(k)}{\partial C(k-1)} + e^*(k)\frac{\partial e(k)}{\partial C(k-1)}\right] \\ &= \frac{1}{2}\Big[e(k)\xi^*(k)\left\{\Phi'(net(k))\right\}^* \mathbf{x}^*(k) \\ &\quad + e^*(k)\xi(k)\Phi'(net(k))\mathbf{x}(k)\Big].\end{aligned} \tag{26}$$

This yields the FANNCGD learning algorithm for nonlinear FIR filters realized as dynamical perceptrons, which is given by

$$\begin{aligned} net(k) &= \mathbf{x}^T(k)\mathbf{w}(k) \\ e(k) &= d(k) - \Phi(net(k)) \\ C(k) &= C(k-1) - \frac{\rho}{2}\Big[e(k)\xi^*(k)\left\{\Phi'(net(k))\right\}^* \mathbf{x}^*(k) \\ &\quad + e^*(k)\xi(k)\Phi'(net(k))\mathbf{x}(k)\Big] \\ \eta(k) &= \frac{1}{\left|\Phi'(net(k))\right|^2 \|\mathbf{x}(k)\|_2^2 + C(k)} \\ \mathbf{w}(k+1) &= \mathbf{w}(k) + \eta(k)e(k)\left(\Phi'[net(k)]\right)^* \mathbf{x}^*(k) \end{aligned} \tag{27}$$

where ρ is the step size of the proposed algorithm and is chosen to be a small positive constant.

IV. Convergence of the FANNCGD Algorithm

The FANNCGD algorithm determines the optimal learning rate $\eta(k)$ for the class of complex-valued nonlinear gradient descent algorithms. Although the FANNCGD algorithm converges in the mean squared error for a range of values ρ and $C(k)$, we can show that for uniform convergence $|e(k)| \to 0$ as $k \to \infty$ by

$$|e(k+1)| \leq |1 - \eta(k)\left|\Phi'(net(k))\right|^2 \|\mathbf{x}(k)\|_2^2 | |e(k)|. \tag{28}$$

For this term to converge in the mean squared error sense, it must stand that

$$|1 - \eta(k)|\Phi'(net(k))|^2 \|\mathbf{x}(k)\|_2^2| \leq 1 \tag{29}$$

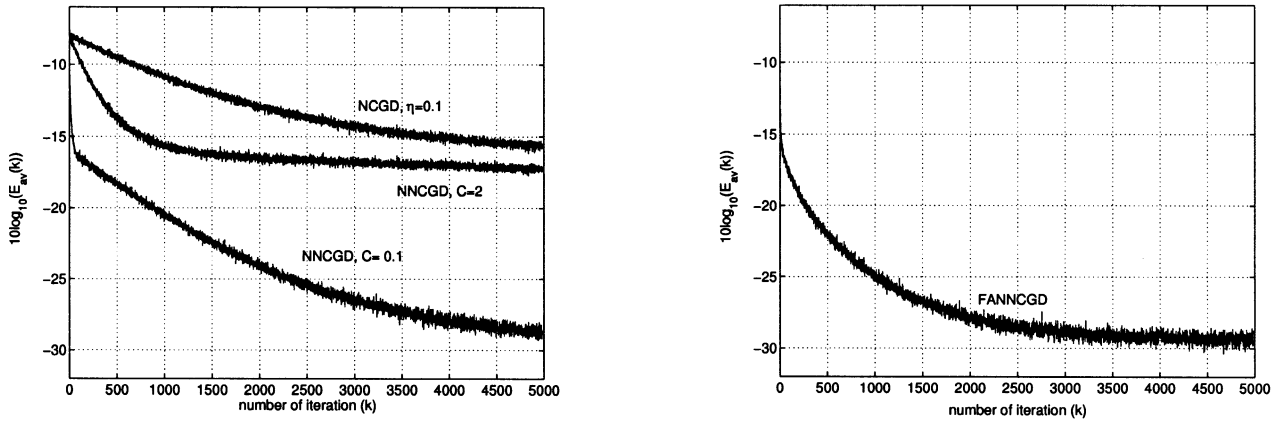
and therefore, the range for $\eta(k)$ becomes

$$0 < \eta(k) < \frac{2}{\left|\Phi'(net(k))\right|^2 \|\mathbf{x}(k)\|_2^2}. \tag{30}$$

$$\begin{aligned}\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)} &= \frac{\partial\left[\Psi^r(k-1))\left(\left|\Phi'(net(k-1))\right|^2 \|\mathbf{x}(k-1)\|_2^2 + C(k-1)\right)^{-1}\right]}{\partial C(k-1)} \\ &= -\frac{\Psi^r(k-1)}{\left[\left|\Phi'(net(k-1))\right|^2 \|\mathbf{x}(k-1)\|^2 + C(k-1)\right]^2}\end{aligned} \tag{23}$$

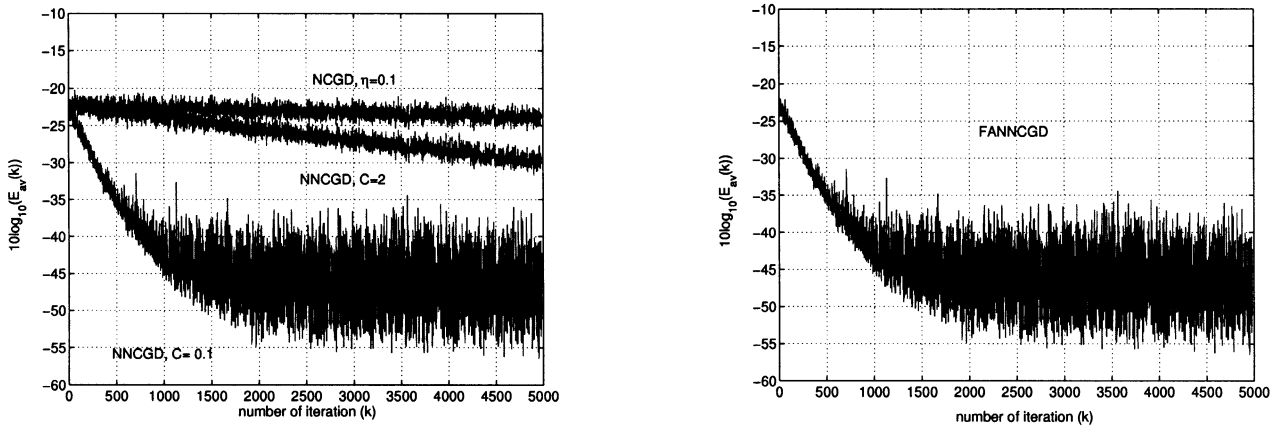
and

$$\begin{aligned}\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)} &= \frac{\partial\left[\Psi^i(k-1)\left(\left|\Phi'(net(k-1))\right|^2 \|\mathbf{x}(k-1)\|_2^2 + C(k-1)\right)^{-1}\right]}{\partial C(k-1)} \\ &= -\frac{\Psi^i(k-1)}{\left[\left|\Phi'(net(k-1))\right|^2 \|\mathbf{x}(k-1)\|_2^2 + C(k-1)\right]^2}\end{aligned} \tag{24}$$



(a) (b)

Fig. 3. Convergence curves of NNCGD and FANNCGD on colored input with the hyperbolic tangent function. (a) Convergence curves for NNCGD on colored input. (b) Convergence curve for FANNCGD on colored input.



(a) (b)

Fig. 4. Convergence curves of NNCGD and FANNCGD on nonlinear input with the hyperbolic tangent function. (a) Convergence curves for NNCGD on nonlinear input. (b) Convergence curve for FANNCGD on nonlinear input.

Substituting in the update term for $\eta(k)$, (27), we can then write

$$0 < \frac{1}{|\Phi'(net(k))|^2 \, ||\mathbf{x}(k)||_2^2 + C(k)} < \frac{2}{|\Phi'(net(k))|^2 \, ||\mathbf{x}(k)||_2^2} \tag{31}$$

and solving for $C(k)$ gives

$$\frac{-|\Phi'(net(k))|^2 \, ||\mathbf{x}(k)||_2^2}{2} < C(k) \tag{32}$$

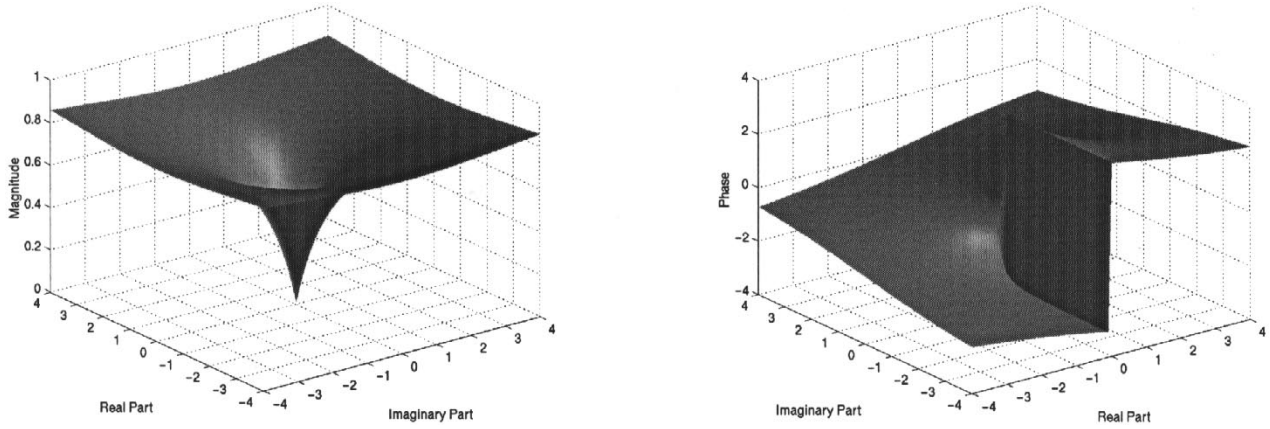
which are the convergence conditions for the FANNCGD algorithm. Convergence analysis for the mean error, mean squared error, and steady state conforms to the analysis in [7].

V. Experimental Results

To investigate the performance of the FANNCGD algorithm compared with the NCGD and NNCGD algorithms, they were all applied to the problem of time-series prediction by averaging the performance curves of 500 independent simulations. For rigour, all algorithms were tested on complex-valued colored and nonlinear input signals with various complex-valued activation functions.

A. Hyperbolic Tangent Function

The algorithms were employed on single neuron FIR complex-valued nonlinear adaptive filters for prediction of colored and nonlinear complex-valued signals. The activation function was the complex-valued hyperbolic tangent function (16), with $\beta = 0.04$ and a tap input of size $N = 10$. The input to all filters was complex-valued white noise $r(k)$ with zero mean and unit variance, which was then passed through a stable AR filter given by (15) for the linear prediction. Fig. 3 shows the performance curves for the NCGD, NNCGD, and FANNCGD algorithms on time series prediction of colored input. The quantitative measure of performance was a logarithmic scale of the averaged cost function $E(k) = 1/2|e(k)|^2$. Fig. 3(a) shows the NCGD algorithm performance curve reaching -15.5 dB with a learning rate $\eta = 0.1$. The NNCGD algorithm performance curve reached -28.8 dB and -17 dB for values of $C = 0.1$ and $C = 2$, respectively. The FANNCGD algorithm [see Fig. 3(b)] converged to -29 dB, which is at least as good at the best choice of C in the NNCGD algorithm. For the second experiment of nonlinear



(a) (b)

Fig. 5. Nonlinear complex-valued activation function $\Phi(z) = z/(\kappa + (1/\epsilon)|z|)$. (a) Magnitude of $\Phi(z) = z/(\kappa + (1/\epsilon)|z|)$. (b) Phase of $\Phi(z) = z/(\kappa + (1/\epsilon)|z|)$.

time series prediction, the input signal $a(k)$ was passed through a benchmark nonlinear filter described by [7]

$$n(k) = \frac{n(k-1)}{1 + n^2(k-1)} + a^3(k) \tag{33}$$

and the nonlinearity in the output neuron was the complex-valued hyperbolic tangent function given in (16) with $\beta = 0.2$. For the task of nonlinear prediction (33), Fig. 4(a) shows the performance curve of the NCGD algorithm reaching -23 dB, and the performance curves of the NNCGD algorithm reaching to -30 dB and -49 dB for values of $C = 2$ and $C = 0.1$, respectively. The performance curve of the FANNCGD algorithm [see Fig. 4(b)] converged to value of -50 dB, which is at least as good as the best performance of C in the NNCGD algorithm.

In Figs. 3 and 4, it is shown that the FANNCGD algorithm reaches the best performance of the NNCGD algorithm when an optimal constant C is chosen. This optimal value of C in the NNCGD algorithm is not known before training, and thus, the FANNCGD algorithm is a robust generalization of the NNCGD algorithm.

The simulation results have shown the FANNCGD algorithm outperforming the NNCGD algorithm for complex-valued linear and nonlinear input signals. It is shown that the NNCGD algorithm can achieve optimal performance given certain input signals for a specific value of C. However, over an averaged number of simulations, the NNCGD algorithm will not obtain as high a global performance as the proposed FANNCGD algorithm.

B. Practical Complex-Valued Activation Function

It is known from Liouville's statement [6] that a function that is analytic and nonlinear cannot be bounded on the entire complex domain. There are many choices of activation functions that satisfy the desirable constraints defined in [6]; however, the proposed FANNCGD algorithm is derived for any complex-valued nonlinear function that satisfies these conditions. To further illustrate this, we employ the frequently used complex-valued function given in [3] and shown in Fig. 5

$$\Phi(z) = \frac{z}{\kappa + \frac{1}{\epsilon}|z|} \tag{34}$$

where κ and ϵ are real positive constants. Although the activation function does not satisfy the Cauchy–Riemann equations, it does satisfy the constraint in [3] and [6] if $\Phi(z) = u + jy$ for some $z \in \mathbb{C}$, where $z = x + jy$; then, if

$$\frac{\partial u}{\partial x}\,\frac{\partial v}{\partial y} \equiv \frac{\partial v}{\partial x}\,\frac{\partial u}{\partial y}$$

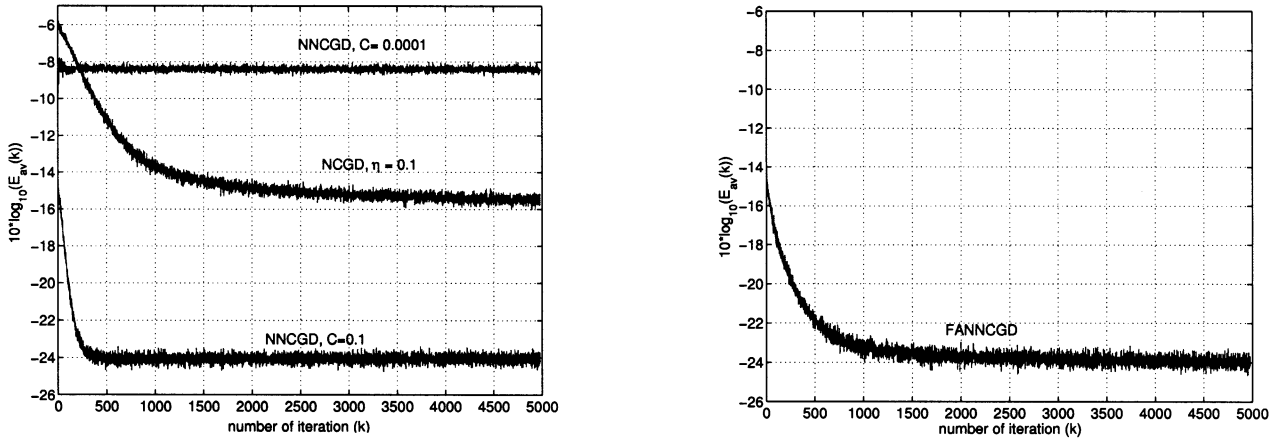
it means that $\Phi(z)$ is not a suitable activation function. This function has the property of mapping a point $z \in \mathbb{C}$ to a unique point $\Phi(z)$ on the open disc $\{z : |z| < \epsilon\}$, and the parameter κ controls the slope of the activation function.

The partial derivatives of (34) are given by [3]

$$\begin{aligned}
u_\nu &= \begin{cases} \left|\frac{\epsilon(\tau^2 + \kappa\epsilon|z|)}{|z|(\kappa\epsilon + |z|)^2}\right|, & \text{if } |z| \neq 0 \\ \frac{1}{c}, & \text{if } |z| = 0 \end{cases} \\
u_\tau &= \begin{cases} -\frac{\epsilon\nu\tau}{|z|(\kappa\epsilon + |z|)^2}, & \text{if } |z| \neq 0 \\ 0, & \text{if } |z| = 0 \end{cases} \\
v_\nu &= \begin{cases} -\frac{\epsilon\nu\tau}{|z|(\kappa\epsilon + |z|)^2}, & \text{if } |z| \neq 0 \\ 0, & \text{if } |z| = 0 \end{cases}
\end{aligned} \tag{35}$$

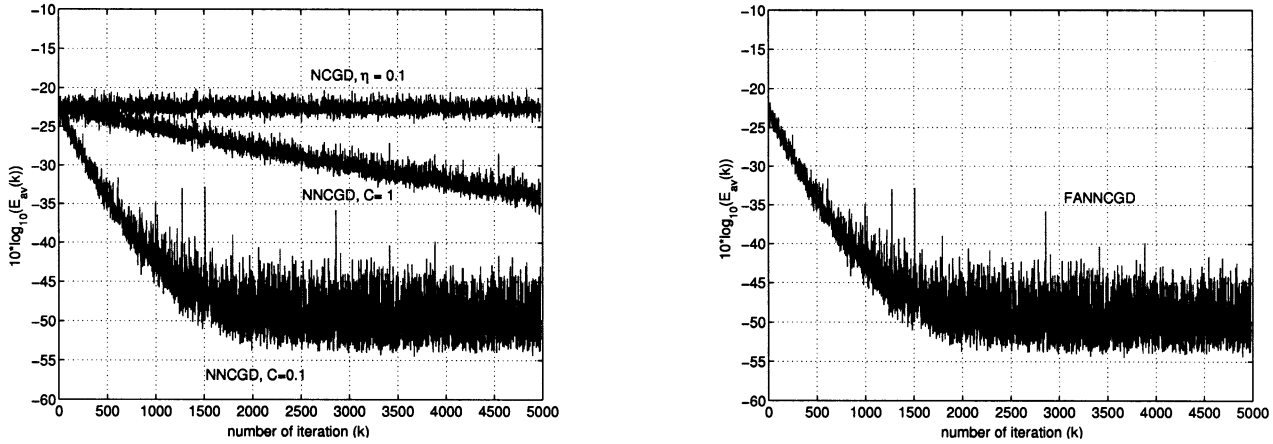
$$v_\tau = \begin{cases} \left|\frac{\epsilon(\tau^2 + \kappa\epsilon|z|)}{|z|(\kappa\epsilon + |z|)^2}\right|, & \text{if } |z| \neq 0 \\ \frac{1}{c}, & \text{if } |z| = 0 \end{cases} \tag{36}$$

where $z = \nu + j\tau$. Fig. 6 shows the performance curves for the NCGD, NNCGD, and FANNCGD algorithms on adaptive prediction of colored input, (15), with $\kappa = 2$ and $\epsilon = 1$. The quantitative measure of performance was a logarithmic scale of the averaged cost function $E(k) = (1/2)|e(k)|^2$. Fig. 6(a) shows the NCGD algorithm performance curve reaching -15.9 dB with a learning rate $\eta = 0.1$. The NNCGD algorithm performance curve reached -24.0 dB and -8.1 dB for values of $C = 0.1$ and $C = 0.0001$, respectively. The FANNCGD algorithm [see Fig. 6(b)] converged to -24 dB, which is at least as good at the best choice of C in the NNCGD algorithm. Fig. 7 shows the performance curves for the NCGD, NNCGD, and FANNCGD algorithms on time series prediction of nonlinear input (33), with $\kappa = 5$ and $\epsilon = 4$. Fig. 7(a) shows the performance curve of the NCGD algorithm reaching -22 dB and the performance curves of the NNCGD algorithm reaching -34 dB and -49 dB for



(a) (b)

Fig. 6. Convergence curves of NNCGD and FANNCGD on colored input with a practical complex-valued activation function (34). (a) Convergence curves for NNCGD on colored output. (b) Convergence curves for FANNCGD on colored output.



(a) (b)

Fig. 7. Convergence curves of NNCGD and FANNCGD on colored input with a practical complex-valued activation function (34). (a) Convergence curves for NNCGD on nonlinear input. (b) Convergence curves for FANNCGD on nonlinear input.

values of $C = 1$ and $C = 0.1$, respectively. The performance curve of the FANNCGD algorithm [see Fig. 7(b)] converged to value of -50 dB, which is at least as good as the best performance of C in the NNCGD algorithm.

VI. Robustness of the FANNCGD Algorithm

With all nonlinear stochastic models, the initial conditions can effect the performance of the systems dramatically. To this cause, an experiment to investigate the robustness of the fully adaptive normalized nonlinear complex gradient descent (FANNCGD) algorithm according to the initial choice of $C(0)$ was carried out on a nonlinear adaptive filter with a single dynamical perceptron using the complex-valued activation function given in (34) as the nonlinearity. The task was time series prediction of complex-valued white noise $r(k)$ that was then passed through the stable AR filter described in (15). The quantitative measure of performance was prediction gain $R_p = 10log_{10}(\hat{\sigma}_s^2/\hat{\sigma}_e^2)$, where $\hat{\sigma}_s^2$ denotes the variance of the expected signal, and $\hat{\sigma}_e^2$ denotes the variance of the prediction error. Fig. 8 shows the effects on the prediction gain for $0 \leq C(0) \leq 1$. The maximum variance of prediction gain for this range of initial conditions is

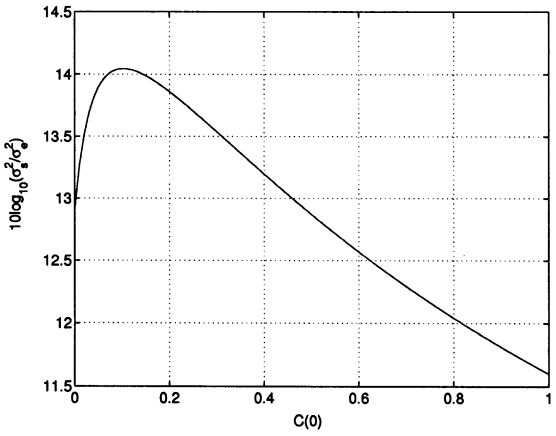


Fig. 8. Prediction gain for varying values of $C(0)$ for FANNCGD.

2.5 dB, which reinforces the robustness of the proposed FANNCGD algorithm with respect to the initial conditions.

VII. Conclusions

A fully adaptive normalized nonlinear complex-valued gradient descent (FANNCGD) algorithm for training nonlinear adaptive filters realized as a dynamical perceptron has been derived. The previously derived real-valued normalized nonlinear gradient descent (NNGD) algorithm has first been extended to manage signals in the complex domain $\mathbb{C}$, resulting in the normalized nonlinear complex-valued gradient descent (NNCGD) algorithm. A fundamental constant term in the derivation of the NNCGD algorithm was made adaptive using a gradient descent-based approach to yield the fully adaptive normalized nonlinear complex-valued gradient descent algorithm. It has been shown that FANNCGD is an improved algorithm to the NNCGD algorithm that optimizes the learning rate by utilizing the Taylor series expansion of the instantaneous output error. The proposed FANNCGD algorithm has been derived for any complex-valued activation function that satisfies the conditions stated in [6]. Experimental results have shown the FANNCGD algorithm outperforming the NNCGD algorithm on colored and nonlinear input signals. It has also been shown that the proposed FANNCGD algorithm is robust to the initial conditions, which compensates for the deficiency in the derivation of the real valued normalized nonlinear gradient descent algorithm.

Appendix A
Derivation of $\partial e(k)/\partial w_n(k)$

The derivation of the Cauchy–Riemann equations state that

$$\frac{\partial e^r(k)}{\partial u(k)}=\frac{\partial e^i(k)}{\partial v(k)},\ \frac{\partial e^r(k)}{\partial v(k)}=-\frac{\partial e^i(k)}{\partial u(k)}$$

and

$$\frac{\partial u(k)}{\partial net^r(k)}=\frac{\partial v(k)}{\partial net^i(k)},\ \frac{\partial v(k)}{\partial net^r(k)}=-\frac{\partial u(k)}{\partial net^i(k)}$$

$$\frac{\partial e(k)}{\partial w_n(k)}=\frac{\partial e^i(k)}{\partial w_n^i(k)}-j\frac{\partial e^r(k)}{\partial w_n^i(k)}. \tag{37}$$

Therefore

$$\begin{aligned}\frac{\partial e^i(k)}{\partial w_n^i(k)}&=\frac{\partial e^i(k)}{\partial v(k)}\left[\frac{\partial v(k)}{\partial net^r(k)}\frac{\partial net^r(k)}{\partial w_n^i(k)}+\frac{\partial v(k)}{\partial net^i(k)}\frac{\partial net^i(k)}{\partial w_n^i(k)}\right]\\&=\frac{\partial e^i(k)}{\partial v(k)}\left[\frac{\partial v(k)}{\partial net^r(k)}\left(-x_n^i(k)\right)+\frac{\partial v(k)}{\partial net^i(k)}\left(x_n^r(k)\right)\right]\end{aligned} \tag{38}$$

and

$$\begin{aligned}\frac{\partial e^r(k)}{\partial w_n^i(k)}&=\frac{\partial e^r(k)}{\partial u(k)}\left[\frac{\partial u(k)}{\partial net^r(k)}\frac{\partial net^r(k)}{\partial w_n^i(k)}+\frac{\partial u(k)}{\partial net^i(k)}\frac{\partial net^i(k)}{\partial w_n^i(k)}\right]\\&=\frac{\partial e^r(k)}{\partial u(k)}\left[\frac{\partial u(k)}{\partial net^r(k)}\left(-x_n^i(k)\right)+\frac{\partial u(k)}{\partial net^i(k)}\left(x_n^r(k)\right)\right]\end{aligned} \tag{39}$$

giving

$$\begin{aligned}\frac{\partial e(k)}{\partial w_n(k)}=&\frac{\partial e^i(k)}{\partial v(k)}\left[\frac{\partial v(k)}{\partial net^r(k)}\left(-x_n^i(k)\right)+\frac{\partial v(k)}{\partial net^i(k)}\left(x_n^r(k)\right)\right]\\&-j\frac{\partial e^r(k)}{\partial u(k)}\left[\frac{\partial u(k)}{\partial net^r(k)}\left(-x_n^i(k)\right)+\frac{\partial u(k)}{\partial net^i(k)}\left(x_n^r(k)\right)\right].\end{aligned} \tag{40}$$

From the above, we have

$$\begin{aligned}\frac{\partial e(k)}{\partial w_n(k)}=&\frac{\partial e^i(k)}{\partial v(k)}\left[\frac{\partial v(k)}{\partial net^r(k)}\left(-x_n^i(k)\right)\right.\\&+\frac{\partial v(k)}{\partial net^i(k)}\left(x_n^r(k)\right)+\frac{\partial u(k)}{\partial net^r(k)}\left(jx_n^i(k)\right)\\&\left.+\frac{\partial u(k)}{\partial net^i(k)}\left(-jx_n^r(k)\right)\right]\\=&-\left[\frac{\partial u(k)}{\partial net^r(k)}\left(x_n^r(k)+jx_n^i(k)\right)\right.\\&\left.+\frac{\partial v(k)}{\partial net^r(k)}\left(-x_n^i(k)+jx_n^r(k)\right)\right]\\=&-x_n(k)\left[\frac{\partial u(k)}{\partial net^r(k)}+j\frac{\partial v(k)}{\partial net^r(k)}\right]\\=&-\Phi'(net(k))x_n(k).\end{aligned} \tag{41}$$

Appendix B
Derivation of $\partial e^*(k)/\partial C(k-1)$

To calculate $\partial e^*(k)/\partial C(k-1)$, we must split it up into its real and imaginary parts to obtain

$$\frac{\partial e^*(k)}{\partial C(k-1)}=\frac{\partial\left(e^*\right)^r(k)}{\partial C(k-1)}+j\frac{\partial\left(e^*\right)^i(k)}{\partial C(k-1)} \tag{42}$$

also knowing that

$$e^*(k)=d^*(k)-\Phi^*[net(k)]=d^*(k)-\Phi^*\left[\mathbf{x}^T(k)\mathbf{w}(k)\right] \tag{43}$$

so it follows that

$$\left(e^*(k)\right)^r=\left(d^*(k)\right)^r-u(k),\quad \left(e^*(k)\right)^i=\left(d^*(k)\right)^i+v(k). \tag{44}$$

Thus

$$\frac{\partial\left(e^*(k)\right)^r}{\partial u(k)}=-1,\quad \frac{\partial\left(e^*(k)\right)^i}{\partial v(k)}=1 \tag{45}$$

to give (46) and (47), shown at the top of the next page. Recognizing that the Cauchy–Riemann equations state

$$\frac{\partial u(k)}{\partial net^r(k)}=\frac{\partial v(k)}{\partial net^i(k)},\quad \frac{\partial v(k)}{\partial net^r(k)}=-\frac{\partial u(k)}{\partial net^i(k)} \tag{48}$$

$$\begin{aligned}\frac{\partial (e^*(k))^r}{\partial C(k-1)} =& \frac{\partial (e^*(k))^r}{\partial u(k)}\left[\frac{\partial u(k)}{\partial net^r(k)}\left(\frac{\partial net_r(k)}{\partial \mathbf{w}^r(k)}\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\frac{\partial net_r(k)}{\partial \mathbf{w}^i(k)}\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\right.\\ &\left.+\frac{\partial u(k)}{\partial net^i(k)}\left(\frac{\partial net_i(k)}{\partial \mathbf{w}^r(k)}\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\frac{\partial net_i(k)}{\partial \mathbf{w}^i(k)}\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\right]\\ =& \frac{\partial u(k)}{\partial net^r(k)}\left(-\mathbf{x}^r(k)\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\mathbf{x}^i(k)\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\\ &+\frac{\partial u(k)}{\partial net^i(k)}\left(-\mathbf{x}^i(k)\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}-\mathbf{x}^r(k)\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\end{aligned} \tag{46}$$

and

$$\begin{aligned}\frac{\partial (e^*(k))^i}{\partial C(k-1)} =& \frac{\partial (e^*(k))^i}{\partial v(k)}\left[\frac{\partial v(k)}{\partial net^r(k)}\left(\frac{\partial net_r(k)}{\partial \mathbf{w}^r(k)}\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\frac{\partial net_r(k)}{\partial \mathbf{w}^i(k)}\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\right.\\ &\left.+\frac{\partial v(k)}{\partial net^i(k)}\left(\frac{\partial net_i(k)}{\partial \mathbf{w}^r(k)}\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\frac{\partial net_i(k)}{\partial \mathbf{w}^i(k)}\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\right]\\ =& \frac{\partial v(k)}{\partial net^r(k)}\left(\mathbf{x}^r(k)\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}-\mathbf{x}^i(k)\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\\ &+\frac{\partial v(k)}{\partial net^i(k)}\left(\mathbf{x}^i(k)\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\mathbf{x}^r(k)\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right).\end{aligned} \tag{47}$$

$$\begin{aligned}\frac{\partial e^*(k)}{\partial C(k-1)} =& \frac{\partial u(k)}{\partial net^r(k)}\left[\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}\left(-\mathbf{x}^r(k)+j\mathbf{x}^i(k)\right)+\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\left(j\mathbf{x}^r(k)+\mathbf{x}^i(k)\right)\right]\\ &+\frac{\partial u(k)}{\partial net^i(k)}\left[\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}\left(-j\mathbf{x}^r(k)-\mathbf{x}^i(k)\right)+\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\left(-\mathbf{x}^r(k)+j\mathbf{x}^i(k)\right)\right]\\ =& -\left[\frac{\partial u(k)}{\partial net^r(k)}\left(\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}-j\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\right.\\ &\left.+\frac{\partial u(k)}{\partial net^i(k)}\left(j\frac{\partial \mathbf{w}^r(k)}{\partial C(k-1)}+\frac{\partial \mathbf{w}^i(k)}{\partial C(k-1)}\right)\right]\mathbf{x}^*(k)\\ =& -\left(\frac{\partial \mathbf{w}(k)}{\partial C(k-1)}\right)^*\left[\frac{\partial u(k)}{\partial net^r(k)}+j\frac{\partial u(k)}{\partial net^i(k)}\right]\mathbf{x}^*(k)\\ =& -\left(\frac{\partial \mathbf{w}(k)}{\partial C(k-1)}\right)^*\left[\Phi'(net(k))\right]^*\mathbf{x}^*(k)\end{aligned} \tag{49}$$

and combining (46) and (47) together yields (49), shown at the top of the page. Using the same techniques in (45) and recognizing (48) yields

$$\frac{\partial e(k)}{\partial C(k-1)} = -\left(\frac{\partial \mathbf{w}(k)}{\partial C(k-1)}\right)\Phi'(net(k))\mathbf{x}(k). \tag{50}$$

ACKNOWLEDGMENT

The authors would like to acknowledge the suggestions and recommendations of the anonymous referees that have contributed to improve the final version of the paper.

REFERENCES

[1] B. Widrow, J. McCool, and M. Ball, "The complex LMS algorithm," *Proc. IEEE*, vol. 63, pp. 719–720, 1975.
[2] A. Hirose, "Continuous complex-valued backpropagation learning," *Electron. Lett.*, vol. 28, no. 20, pp. 1854–1855, 1992.
[3] G. M. Georgiou and C. Koutsougeras, "Complex domain backpropagation," *IEEE Trans. Circuits Syst. II*, vol. 39, pp. 330–334, May 1992.
[4] A. Hirose, "Proposal of fully complex-valued neural networks," in *Proc. IV JCNN*, Baltimore, MD, 1992, pp. 152–157.
[5] N. Benvenuto and F. Piazza, "On the complex backpropagation algorithm," *IEEE Trans. Signal Processing*, vol. 40, pp. 967–969, Apr. 1992.
[6] T. Kim and T. Adali, "Approximation by fully complex MLP using elementary transcendental activation functions," in *Proc. XI IEEE Workshop Neural Networks Signal Process.*, 2001, pp. 203–212.
[7] D. Mandic and J. Chambers, *Recurrent Neural Networks for Prediction*. Chichester, U.K.: Wiley, 2001.

[8] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 1996.
[9] S. A. Billings, H. B. Jamaluddin, and S. Chen, "Properties of neural networks with applications to modeling nonlinear dynamical systems," *Int. J. Contr.*, vol. 55, pp. 193–224, 1992.
[10] A. I. Hanna, D. P. Mandic, and M. Razaz, "A normalized backpropagation learning algorithm for multilayer feed-forward neural adaptive filters," in *Proc. XI IEEE Workshop Neural Networks Signal Process.*, 2001, pp. 63–72.
[11] D. P. Mandic and J. A. Chambers, "Toward the optimal learning rate for backpropagation," *Neural Process. Lett.*, vol. 11, no. 1, pp. 1–5, 2000.
[12] V. J. Mathews and Z. Xie, "Stocastic gradient adaptive filters with gradient adaptive step sizes," *IEEE Trans. Signal Processing*, vol. 41, pp. 2075–2087, June 1993.
[13] W. Ang and B. Farhang-Boroujeny, "A new class of gradient adaptive step-size LMS algorithms," *IEEE Trans. Signal Processing*, vol. 49, pp. 805–810, Apr. 2001.
[14] A. M. Kuzminskiy, "Self-adjustment of an adaptation coefficient of a noise compensator in a nonstationary process," *Izvestiya VUZ. Radioelektronika*, vol. 29, no. 3, pp. 103–105, 1986.
[15] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295–307, 1988.
[16] A. Cichocki, R. Unbenhauen, P. Signal, and Sons, *Neural networks for optimization and Signal Processing*. New York: Wiley, 1996.

Andrew Ian Hanna was born in Blackpool, Lancashire, U.K. He received the B.Sc. degree in computer science from the University of East Anglia, Norwich, Norfolk, U.K., in 1998. He is currently pursuing the Ph.D. degree with the School of Computing Sciences at the University of East Anglia.

His current research areas include nonlinear adaptive filtering, neural networks, two-dimensional adaptive filtering, and atomic force microscopy image restoration.

Danilo P. Mandic (M'99) received the Ph.D. degree in nonlinear adaptive signal processing in 1999 from Imperial College, London, London, U.K.

He is a Senior Lecturer with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K. He has previously taught at the Universities of East Anglia, Norwich, Norfolk, U.K., and Banja Luka, Bosnia Herzegovina. He has written over 100 publications on a variety of aspects of signal processing and a research monograph on recurrent neural networks. He has been a Guest Professor at the Catholic University Leuven, Leuven, Belgium, and Frontier Researcher at the Brain Science Institute RIKEN, Tokyo, Japan.

Dr. Mandic is a Member of the IEEE Signal Processing Society Technical Committee on Neural Networks for Signal Processing, Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II, and Associate Editor for *International Journal of Mathematical Modeling and Algorithms*. He has won awards for his papers and for the products coming from his collaboration with industry.