Neural Networks 23 (2010) 426–434

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Split quaternion nonlinear adaptive filtering

Bukhari Che Ujang *, Clive Cheong Took, Danilo P. Mandic

Communications and Signal Processing Research Group, Department of Electrical and Electronic Engineering, Imperial College London, South Kensington Campus, London SW7 2AZ, UK

ARTICLE INFO

Article history:
Received 29 June 2009
Received in revised form 13 October 2009
Accepted 22 October 2009

Keywords:
Quaternion-valued adaptive filters
Nonlinear adaptive filtering
Cauchy–Riemann–Fueter equation
Quaternion Multilayer Perceptron
Wind modelling

ABSTRACT

A split quaternion learning algorithm for the training of nonlinear finite impulse response adaptive filters for the processing of three- and four-dimensional signals is proposed. The derivation takes into account the non-commutativity of the quaternion product, an aspect neglected in the derivation of the existing learning algorithms. It is shown that the additional information taken into account by a rigorous treatment of quaternion algebra provides improved performance on hypercomplex processes. A rigorous analysis of the convergence of the proposed algorithms is also provided. Simulations on both benchmark and real-world signals support the approach.

1. Introduction

Recent advances in sensor technology and the increasing number of corresponding applications (sensor networks, data fusion) have highlighted the need for efficient multidimensional adaptive signal processing algorithms to efficiently deal with such multidimensional data. To ensure optimal processing power, instead of treating each dimension separately, the signal needs to be processed as a whole, thus fully exploiting the available information. A number of machine learning algorithms have been developed to fulfill this objective, especially in the area of neural networks (Arena, Fortuna, Muscato, & Xibilia, 1998; Nitta, 1993; Nitta & de Garis, 1992). One convenient approach is to represent multidimensional signals as vectors in $\mathbb{R}^n$, leading to one of the first multidimensional learning algorithms for neural networks, the Three-Dimensional Vector Back-Propagation (3DV-BP) (Nitta & de Garis, 1992). However, there are also problems associated with vectorial approaches, for instance, the matrix operation in Nitta and de Garis (1992) does not fully exploit the coupling between the three dimensions. Improvements include the Vector Product Back-Propagation (VP-BP), which addresses this issue through the use of vector products (Nitta, 1993). However, this algorithm cannot update the weights in the presence of non-zero error. Both these algorithms use a split quaternion logistic activation function whereby each component is processed independently. The universal function approximation capabilities for both algorithms has not been investigated as no density theorem has been proved for real vector spaces (Arena et al., 1998, pp 67–71).

When dealing with three- and four-dimensional signals such as signals in optics,vector fields, and three-dimensional motion data, it is natural to consider the processing in the quaternion domain $\mathbb{H}$. Quaternions were first conceived by Hamilton in 1843, and have found applications in robotics (Biamino, Cannata, Maggiali, & Piazza, 2005), molecular modelling (Karney, 2007), and computer graphics (Choe & Faraway, 2004). The dilemma of the modelling in the quaternion domain versus the modelling in $\mathbb{R}^4$ has been long present (Heaviside, 1893; MacFarlane, 1893; Silva & Martins, 2002) and traditionally, quaternion based adaptive signal processing has not been as prominent as that based on vectors.

In the area of neural networks, quaternion adaptive signal processing started with the introduction of the Quaternion-valued Multilayer Perceptron (QMLP) and the benefits stemming from the algebraic properties of the quaternion domain led to an enhanced performance over previous algorithms of this kind (Arena et al., 1998). Although the QMLP applies a split quaternion logistic function similar to that used in the previous algorithms, unlike the vectorial approaches in $\mathbb{R}^3$ and $\mathbb{R}^4$, its approximation capabilities are well understood as the density theorem in $\mathbb{H}$ has been proven (Arena et al., 1998, pp 67–71). One of the applications of QMLP was in polarized signal classification (Buchholz & Bihan, 2008). Despite the satisfactory results obtained, the algorithm for the training of QMLP ignores the non-commutativity of the quaternion product and cannot reach theoretical performance limits.

Recently, for adaptive filtering of three- and four-dimensional signals, the Least Mean Square (LMS) algorithm has been extended to the quaternion domain, with the introduction of the Quaternion Least Mean Square (QLMS) algorithm (Cheong-Took & Mandic, 2009). The QLMS algorithm exhibits a superior performance over

* Corresponding author. Tel.: +44 (0) 20 7594 6310; fax: +44 (0) 20 7594 6302.
E-mail addresses: che.che-ujang07@imperial.ac.uk (B.C. Ujang), c.cheong-took@imperial.ac.uk (C.C. Took), d.mandic@imperial.ac.uk (D.P. Mandic).

0893-6080/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2009.10.006

the standard multichannel LMS algorithm for the filtering of hypercomplex processes, as the use of quaternion algebra naturally models the coupling between the components (Cheong-Took & Mandic, 2009). However, the linear nature of the quaternion finite impulse response (FIR) filter renders the QLMS algorithm suboptimal for the processing of nonlinear multidimensional signals (Arena et al., 1998; Mitsubori & Saito, 1994). To this end, we here introduce a "split quaternion" nonlinear adaptive filtering algorithm.

The aim of this article is to introduce a three- and four-dimensional valued nonlinear FIR adaptive filter suitable for the processing of nonlinear signals and modelling of nonlinear systems. Due to a number of open issues in the analysis of nonlinear quaternion-valued functions, it is difficult to extend the corresponding approaches from $\mathbb{C}$ to $\mathbb{H}$, as the only analytic quaternion function is a linear quaternion function (Sudbery, 1979). We, therefore, resort to applying nonlinearities component-wise. The learning algorithms introduced, the Split Quaternion Nonlinear Adaptive Filtering Algorithm (SQAFA) and the Adaptive Amplitude Split Quaternion Nonlinear Adaptive Filtering Algorithm (AASQAFA), are derived rigorously in order to explicitly address the non-commutativity of the quaternion product and to tackle the problems associated with a large dynamical range of the quaternion signal.

The article is organized as follows. In Section 2, we briefly review the basics of quaternion algebra. This is followed by a brief discussion of the analyticity of nonlinear functions in $\mathbb{H}$. The proposed SQAFA and AASQAFA are derived in Section 4. Section 5 analyzes the convergence of the proposed algorithms. The performances of SQAFA algorithm and AASQAFA algorithm are compared against QMLP and the corresponding complex and multidimensional real-valued algorithms, through comprehensive simulations on both benchmark and real-world multidimensional data. Section 6 gives a discussion of the results obtained. Finally, the article concludes in Section 7.

2. Nonlinear functions in $\mathbb{H}$

A basic quaternion variable $q \in \mathbb{H}$ has a scalar part, or equivalently the real part, and a vector part comprising three imaginary parts, which can be represented as

$$\begin{aligned} q &= [q_a, \mathbf{q}] \\ &= q_a + q_b\imath + q_c\jmath + q_d\kappa \end{aligned} \tag{1}$$

where $q_a, q_b, q_c, q_d \in \mathbb{R}$ and $\imath, \jmath, \kappa$ are orthogonal unit vectors.

The relationship between the orthogonal unit vectors $\imath, \jmath, \kappa$ are

$$\begin{aligned} &\imath\jmath = \kappa; \qquad \jmath\kappa = \imath; \qquad \kappa\imath = \jmath; \\ &\imath\jmath\kappa = \imath^2 = \jmath^2 = \kappa^2 = -1 \end{aligned} \tag{2}$$

A quaternion operation which needs to be carefully considered is the multiplication, given by

$$\begin{aligned} wx &= [w_a, \mathbf{w}][x_a, \mathbf{x}] \\ &= [w_a x_a - \mathbf{w}\cdot\mathbf{x},\ w_a\mathbf{x} + x_a\mathbf{w} + \mathbf{w}\times\mathbf{x}] \end{aligned} \tag{3}$$

where the symbols "$\cdot$" and "$\times$" denote, respectively, to the dot-product and cross-product. The non-commutativity of the quaternion product $wx \neq xw$, arises due to the presence of the outer product. Similar to the complex case, the conjugate of a quaternion q is $q^* = [q_a, \mathbf{q}]^* = [q_a, -\mathbf{q}]$, and the norm $\|q\|_2^2 = qq^*$. From here onwards, all quantities are treated as quaternion valued, unless stated otherwise.

The choice of a quaternion-valued nonlinearity in adaptive signal processing is an open issue. In order for a nonlinear function to be analytic in the quaternion domain, it must satisfy the Cauchy–Riemann–Fueter (CRF) equation, which is an extension of the Cauchy–Riemann equation in $\mathbb{C}$ (Sudbery, 1979).

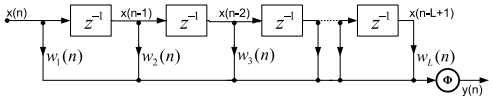


Fig. 1. Nonlinear adaptive finite impulse response (FIR) filter.

For a complex function $f(z) = u(x, y) + v(x, y)\imath$, the Cauchy–Riemann conditions are given by

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}; \qquad \frac{\partial v}{\partial x} = -\frac{\partial u}{\partial y} \tag{4}$$

that is, for a complex function $f(z)$ to be analytic in $\mathbb{C}$, we have

$$\frac{\partial f}{\partial x} + \frac{\partial f}{\partial y}\imath = 0. \tag{5}$$

In practice, conditions (4) are very convenient and simplify the derivation of learning algorithms for nonlinear adaptive filtering in $\mathbb{C}$ (Mandic & Goh, 2009).

On the other hand, the CRF condition in $\mathbb{H}$ states that for a function $f(q)$ to be analytic in $\mathbb{H}$ (Sudbery, 1979), it must satisfy

$$\frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\imath + \frac{\partial f}{\partial y}\jmath + \frac{\partial f}{\partial z}\kappa = 0 \tag{6}$$

where $q = t + x\imath + y\jmath + z\kappa$.

However, in order to provide a genuine extension of nonlinear adaptive filtering algorithms from $\mathbb{C}$ to $\mathbb{H}$, we should have a nonlinearity that satisfies the CRF condition. This is not straightforward; for example, applying the CRF equation (6) to the elementary transcendental tanh function gives

$$\frac{\partial \tanh(q)}{\partial q} \neq 0. \tag{7}$$

Thus, the tanh function is not analytic, rendering it unsuitable for a nonlinearity within a nonlinear quaternion-valued adaptive filter.

To design quaternion-valued feedforward adaptive filters, and yet to maintain the nonlinearity of architectures, analogous to split complex filtering in $\mathbb{C}$ (Georgiou & Koutsougeras, 1992; Kim & Adali, 2003), we propose to use "split quaternion" nonlinear function, where a real nonlinearity is applied independently to each component of the quaternion-valued signal. Since the nonlinear tanh function employed as the nonlinear function is odd-symmetric, that is, $\Phi\left(-\mathbf{w}^T(n)\mathbf{x}(n)\right) = -\Phi\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)$, this odd-symmetry yields

$$\Phi'^*\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) = \Phi'\left(\mathbf{x}^H(n)\mathbf{w}^*(n)\right) \tag{8}$$

which is exploited in the derivation of learning algorithms for quaternion-valued nonlinear filters; symbols $(\cdot)^H$ and $(\cdot)^T$ denote the Hermitian and vector transpose operator.

Due to its component-wise operation, the proposed "split quaternion" approach is not analytic in $\mathbb{H}$ as it does not satisfy the CRF condition (6). However, the proposed approach is analytic component-wise and is bounded, making it suitable as a nonlinearity in neural architectures.

3. Nonlinear adaptive filtering in $\mathbb{H}$

In this section, we highlight the problems associated with the existing quaternion-valued learning algorithms for nonlinear filters, and provide rigorous derivation of the proposed algorithms, which fully utilize the available second order statistical information. The architecture of a nonlinear adaptive FIR filter considered is given in Fig. 1.

The standard cost function in quaternion-valued adaptive filtering is given by

$$E(n) = \mathrm{e}_a^2(n) + \mathrm{e}_b^2(n) + \mathrm{e}_c^2(n) + \mathrm{e}_d^2(n) \quad (9)$$

$$= e(n)\mathrm{e}^*(n) \quad (10)$$

where the error $e(n) = d(n) - y(n)$ and $y(n) = \Phi(s(n))$, with $d(n)$, $y(n)$, and $\Phi(\cdot)$ denoting, respectively, the desired signal, output signal and split quaternion nonlinear function. The "net" input $s(n)$ is defined as $s(n) = \mathbf{w}^T(n)\mathbf{x}(n)$, where $\mathbf{w}(n)$ and $\mathbf{x}(n)$ correspond, respectively, to the adaptive weight vector and the filter input, and symbols $(\cdot)^T$ and $(\cdot)^*$ denote the transpose and quaternion conjugate operator. Analogous to the split complex approach, whereby each component is processed independently (Leung & Haykin, 1991), the split quaternion nonlinear function is defined as

$$\Phi(q) = \Phi_a(q) + \Phi_b(q)\imath + \Phi_c(q)\jmath + \Phi_d(q)\kappa \quad (11)$$

where Φ_a is a real-valued nonlinear activation function applied to the real part of the quaternion q_a, Φ_b to the $\imath$ part q_b, Φ_c to the $\jmath$ part q_c, and Φ_d to the κ part q_d.

Thus, the output of the adaptive filter considered in this work is given by

$$y(n) = \Phi_a(\mathbf{w}^T(n)\mathbf{x}(n)) + \Phi_b(\mathbf{w}^T(n)\mathbf{x}(n))\imath + \Phi_c(\mathbf{w}^T(n)\mathbf{x}(n))\jmath + \Phi_d(\mathbf{w}^T(n)\mathbf{x}(n))\kappa. \quad (12)$$

3.1. Derivation of the learning algorithm for FIR quaternion- valued nonlinear adaptive filters

The standard learning algorithm for QMLP minimizes the cost function (9), through a gradient descent update of the coefficients, given by $\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\nabla_{\mathbf{w}}E(n)$. When this algorithm is applied to a single perceptron (nonlinear FIR filter considered in this work), we have the error gradient given by (Arena et al., 1998)

$$\nabla_{\mathbf{w}}E(n) = 2e(n)\frac{\mathrm{d}\mathrm{e}^*(n)}{\mathrm{d}\mathbf{w}(n)} = -e(n)\underbrace{\Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\big(-2\mathbf{x}^*(n)\big)}_{\nabla_{\mathbf{w}}y(n)} \quad (13)$$

where $\Phi'(\cdot)$ denotes the derivative of the activation function $\Phi(\cdot)$ (Fig. 1) with respect to $\mathbf{s}(n)$. However, if the non-commutativity of the quaternion product is considered as in our proposed algorithms, the error gradient becomes

$$\nabla_{\mathbf{w}}E(n) = e(n)\frac{\mathrm{d}\mathrm{e}^*(n)}{\mathrm{d}\mathbf{w}(n)} + \frac{\mathrm{d}e(n)}{\mathrm{d}\mathbf{w}(n)}\mathrm{e}^*(n). \quad (14)$$

Comparing (13) with (14), the additional term $\frac{\mathrm{d}e(n)}{\mathrm{d}\mathbf{w}(\mathbf{n})}\mathrm{e}^*(n)$ in (14) helps to make full use of the available second order statistical information. From (13), it is clear that $\nabla_{\mathbf{w}}E(n)$ is also a function of $\nabla_{\mathbf{w}}y(n)$ (the derivation is given in Appendix), that is

$$\nabla_{\mathbf{w}}y(n) = \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\big(-2\mathbf{x}^*(n)\big) \quad (15)$$

This yields the weight update

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\Big(e(n)\Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\mathbf{x}^*(n)\Big). \quad (16)$$

Hence, we propose to calculate the gradient of $E(n)$ (see Appendix) based on

$$\nabla_{\mathbf{w}}E(n) = e(n)\frac{\mathrm{d}\mathrm{e}^*(n)}{\mathrm{d}\mathbf{w}(n)} + \frac{\mathrm{d}e(n)}{\mathrm{d}\mathbf{w}(n)}\mathrm{e}^*(n) = -\Big(e(n)\nabla_{\mathbf{w}}y^*(n) + \nabla_{\mathbf{w}}y(n)\mathrm{e}^*(n)\Big) \quad (17)$$

which clearly demonstrates that the gradient $\nabla_{\mathbf{w}}E(n)$ is also a function of $\nabla_{\mathbf{w}}y^*(n)$. Similarly, as in (15), it can be shown that

$$\nabla_{\mathbf{w}}y^*(n) = \Phi'\big(\mathbf{x}^H(n)\mathbf{w}^*(n)\big)\big(4\mathbf{x}^*(n)\big). \quad (18)$$

This was not considered in previously introduced algorithms for quaternion-valued filter (Arena et al., 1998).

For mathematical rigour, it is, therefore, crucial to consider the expression (14) instead of (13), in order to comply with the non-commutativity of the quaternion product.

3.2. Derivation of the split quaternion adaptive filtering algorithm (SQAFA)

Based on the properties of quaternion algebra, the SQAFA for FIR adaptive filters is derived based on the cost function (10), which can be expressed as

$$E(n) = \Big(d(n) - y(n)\Big)\Big(d^*(n) - y^*(n)\Big) = d(n)d^*(n) - d(n)y^*(n) - y(n)d^*(n) + y(n)y^*(n). \quad (19)$$

The error gradient of (19) can now be calculated from

$$\nabla_{\mathbf{w}}E(n) = -d(n)\nabla_{\mathbf{w}}y^*(n) - \nabla_{\mathbf{w}}y(n)d^*(n) + y(n)\nabla_{\mathbf{w}}y^*(n) + \nabla_{\mathbf{w}}y(n)y^*(n). \quad (20)$$

Replacing the expression for the gradient $\nabla_{\mathbf{w}}y(n)$ from (15) and $\nabla_{\mathbf{w}}y^*(n)$ from (18) into (20) yields

$$\nabla_{\mathbf{w}}E(n) = -4e(n)\Phi'\big(\mathbf{x}^H(n)\mathbf{w}^*(n)\big)\mathbf{x}^*(n) + 2\Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\mathbf{x}^*(n)\mathrm{e}^*(n). \quad (21)$$

Finally, the weight update for the SQAFA for the training of quaternion- valued nonlinear adaptive filters can be expressed as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\Big(2e(n)\Phi'\big(\mathbf{x}^H(n)\mathbf{w}^*(n)\big)\mathbf{x}^*(n) - \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\mathbf{x}^*(n)\mathrm{e}^*(n)\Big) \quad (22)$$

where μ is a real-valued learning rate. Observe the difference between the weight update in (16) and weight update in (22). Due to the rigorous use of quaternion algebra in (22), we also have the conjugate terms, which help to capture complete second order statistics.

3.3. Derivation of adaptive amplitude split quaternion adaptive filtering algorithm (AASQAFA)

Architectures with fixed nonlinearities are not suitable for real-world signals with large dynamical range. To cope with the large dynamics of a signal, we can introduce an adaptive slope of the activation function. However, the adaptive slope of the activation function is interchangeable with the time varying step size of the learning algorithm, rendering it less effective (Mandic & Chambers, 1999). In order to circumvent this problem, we can use a trainable amplitude of the activation function (Trentin, 2001). The same concept was applied to nonlinear FIR adaptive filter in $\mathbb{R}$ (Hanna & Mandic, 2002) and then extended to the recurrent neural network for processing in $\mathbb{C}$ (Goh & Mandic, 2003). These algorithms have significantly superior performance compared with their counterparts with fixed nonlinearities. We shall now introduce a trainable amplitude activation function into the SQAFA, termed the Adaptive Amplitude Split Quaternion Adaptive Filtering Algorithm (AASQAFA).

An adaptive amplitude of nonlinearity can be introduced as (Trentin, 2001)

$$\Phi\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) = \lambda(n)\overline{\Phi}\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \tag{23}$$

where $\lambda(n)$ denotes the time varying amplitude and $\overline{\Phi}(\cdot)$ the real nonlinearity with unit amplitude applied component-wise. In the context of "split quaternion" filtering, this can be formulated as

$$\Phi\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) = \lambda_a(n)\bar{\Phi}_a\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) + \lambda_b(n)\bar{\Phi}_b\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\imath + \lambda_c(n)\bar{\Phi}_c\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\jmath + \lambda_d(n)\bar{\Phi}_d\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\kappa \tag{24}$$

where $\lambda_a(n)$ is the amplitude of the nonlinearity for the real part of the quaternion, $\lambda_b(n)$ for the $\imath$ part, $\lambda_c(n)$ for the $\jmath$ part and $\lambda_d(n)$ for the κ part.

The update of the adaptive amplitude is derived based on

$$\lambda(n+1) = \lambda(n) - \rho\nabla_\lambda E(n) \tag{25}$$

where ρ is a real-valued learning rate.

The error gradient $\nabla_\lambda E(n)$ is given as

$$\begin{aligned}\nabla_\lambda E(n) &= \frac{\partial E(n)}{\partial\lambda(n)} = \frac{\partial\left[e(n)e^*(n)\right]}{\partial\lambda(n)} \\ &= e(n)\frac{\partial e^*(n)}{\partial\lambda(n)} + \frac{\partial e(n)}{\partial\lambda(n)}e^*(n).\end{aligned} \tag{26}$$

From (24), since each dimension is treated separately, it is convenient to define the corresponding component-wise errors as

$$\begin{aligned}e_a(n) &= d_a(n) - \lambda_a(n)\bar{\Phi}_a\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \\ e_b(n) &= d_b(n) - \lambda_b(n)\bar{\Phi}_b\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \\ e_c(n) &= d_c(n) - \lambda_c(n)\bar{\Phi}_c\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \\ e_d(n) &= d_d(n) - \lambda_d(n)\bar{\Phi}_d\left(\mathbf{w}^T(n)\mathbf{x}(n)\right).\end{aligned} \tag{27}$$

As the adaptive amplitude is applied component-wise, the error gradient $\nabla_\lambda E(n)$ for each dimension can be optimized separately. For instance, the error gradient with respect to λ_a, $\nabla_{\lambda_a}E(n)$ is given as

$$\begin{aligned}\nabla_{\lambda_a}E(n) &= e_a(n)\frac{\partial e_a^*(n)}{\partial\lambda_a(n)} + \frac{\partial e_a(n)}{\partial\lambda_a(n)}e_a^*(n) \\ &= -2e_a(n)\bar{\Phi}_a\left(\mathbf{w}^T(n)\mathbf{x}(n)\right).\end{aligned} \tag{28}$$

The updates for the amplitudes of all the four nonlinearities are, therefore, given by

$$\lambda_a(n+1) = \lambda_a(n) + \rho e_a(n)\bar{\Phi}_a\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \tag{29}$$

$$\lambda_b(n+1) = \lambda_b(n) + \rho e_b(n)\bar{\Phi}_b\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \tag{30}$$

$$\lambda_c(n+1) = \lambda_c(n) + \rho e_c(n)\bar{\Phi}_c\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \tag{31}$$

$$\lambda_d(n+1) = \lambda_d(n) + \rho e_d(n)\bar{\Phi}_d\left(\mathbf{w}^T(n)\mathbf{x}(n)\right). \tag{32}$$

4. Convergence analysis of SQAFA and AASQAFA

This section provides a rigorous account of the convergence of the proposed algorithms. This is achieved based upon the relationship between the a priori, and a posteriori error, and by deriving the stepsize bound which ensures convergence. Following the approach from Soria-Olivas, Maravilla, Guerrero-Martinez, Martinez-Sober, and Espi-Lopez (1998) and Mandic and Chambers (2001), consider the first order Taylor series expansion

$$\|\bar{e}(n)\|_2^2 = \|\tilde{e}(n)\|_2^2 + \Delta\mathbf{w}^H(n)\frac{\partial\|\tilde{e}(n)\|_2^2}{\partial\mathbf{w}(n)} \tag{33}$$

where $\bar{e}(n)$, $\tilde{e}(n)$, $\Delta\mathbf{w}^H(n)$ and $\frac{\partial\|\tilde{e}(n)\|_2^2}{\partial\mathbf{w}(n)}$ are, respectively, the a posteriori error, the a priori error, the Hermitian of the weight update and the error gradient.[1] The a posteriori output error $\bar{e}$ and the a priori output error $\tilde{e}$ are defined as

$$\bar{e}(n) = d(n) - \Phi\left(\mathbf{w}^T(n+1)\mathbf{x}(n)\right) \tag{34}$$

$$\tilde{e}(n) = d(n) - \Phi\left(\mathbf{w}^T(n)\mathbf{x}(n)\right) \tag{35}$$

In order for the filter to converge, the a priori and the a posteriori errors need to satisfy

$$\|\bar{e}(n)\|_2^2 < \|\tilde{e}(n)\|_2^2. \tag{36}$$

In the following analysis, we shall make two standard assumptions; (i) the learning rate μ is small; (ii) at convergence the principle of orthogonality applies, that is, $\tilde{e}(n)$ is statistically independent of $\mathbf{x}(n)$ (Haykin, 2002).

4.1. Convergence of SQAFA

The term $\Delta\mathbf{w}^H(n)$ in (33) is obtained by applying the Hermitian transpose operator to (22) giving

$$\begin{aligned}\Delta\mathbf{w}^H = \mu\big[&2\mathbf{x}^T(n)\Phi'^*\left(\mathbf{x}^H(n)\mathbf{w}^*(n)\right)\tilde{e}^*(n) \\ &- \tilde{e}(n)\mathbf{x}^T(n)\Phi'^*\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\big]\end{aligned} \tag{37}$$

The term $\frac{\partial\|\tilde{e}(n)\|_2^2}{\partial\mathbf{w}(n)}$ is the error gradient of SQAFA in (21) and is given by

$$\begin{aligned}\frac{\partial\|\tilde{e}(n)\|_2^2}{\partial\mathbf{w}(n)} = -\big[&4\tilde{e}(n)\Phi'\left(\mathbf{x}^H(n)\mathbf{w}^*(n)\right)\mathbf{x}^*(n) \\ &- 2\Phi'\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\mathbf{x}^*(n)\tilde{e}^*(n)\big]\end{aligned} \tag{38}$$

Substituting (37) and (38) into the Taylor series expansion (33) gives

$$\begin{aligned}\|\bar{e}(n)\|_2^2 = \|\tilde{e}(n)\|_2^2 - \mu\Big(&\big[2\mathbf{x}^T(n)\Phi'^*\left(\mathbf{x}^H(n)\mathbf{w}^*(n)\right)\tilde{e}^*(n) \\ &- \tilde{e}(n)\mathbf{x}^T(n)\Phi'^*\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\big]\big[4\tilde{e}(n)\Phi'\left(\mathbf{x}^H(n)\mathbf{w}^*(n)\right)\mathbf{x}^*(n) \\ &- 2\Phi'\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\mathbf{x}^*(n)\tilde{e}^*(n)\big]\Big).\end{aligned} \tag{39}$$

Enforcing the orthogonality condition to (39) and factorizing the term $\|\tilde{e}(n)\|_2^2$ yields

$$\|\bar{e}(n)\|_2^2 = \|\tilde{e}(n)\|_2^2\Big[1 - 10\mu\mathbf{x}^T(n)\mathbf{x}^*(n)\|\Phi'\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\|_2^2\Big]. \tag{40}$$

In order to satisfy the convergence condition (36), from (40) we have

$$0 < 10\mu\mathbf{x}^T(n)\mathbf{x}^*(n)\|\Phi'\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\|_2^2 < 1. \tag{41}$$

Solving for μ finally gives the range of s the stepsize which ensures stability of SQAFA

$$0 < \mu < \frac{1}{10\mathbf{x}^T(n)\mathbf{x}^*(n)\|\Phi'\left(\mathbf{w}^T(n)\mathbf{x}(n)\right)\|_2^2}. \tag{42}$$

[1] Observe that the higher order derivatives in the Taylor series expansion vanish.

4.2. Convergence of AASQAFA

In the case of AASQAFA, each parameter λ controls the amplitude of the nonlinearity in their respective dimension, hence, the convergence analysis is conducted separately for each dimension. In order for AASQAFA to converge, $\lambda_a(n)$, $\lambda_b(n)$, $\lambda_c(n)$ and $\lambda_d(n)$ must each converge. We shall illustrate the analysis based on the convergence for $\lambda_a(n)$.

The a priori error in the real part $\tilde{e}_a(n)$, and the a posteriori error in the real part $\bar{e}_a(n)$, are given by

$$\tilde{e}_a(n) = d_a(n) - \lambda_a(n)\bar{\Phi}_a\big(\mathbf{w}^T(n)\mathbf{x}(n)\big) \tag{43}$$

$$\bar{e}_a(n) = d_a(n) - \lambda_a(n)\bar{\Phi}_a\big(\mathbf{w}^T(n+1)\mathbf{x}(n)\big). \tag{44}$$

Since λ_a corresponds to the real part of a quaternion quantity, we shall consider only the real part of the Taylor series expansion. From (33), we have

$$\|\bar{e}_a(n)\|_2^2 = \|\tilde{e}_a(n)\|_2^2 + \Delta_a\mathbf{w}^H(n)\frac{\partial\|\tilde{e}_a(n)\|_2^2}{\partial\mathbf{w}(n)} \tag{45}$$

where the term $\Delta_a\mathbf{w}^H(n)$ refers to the Hermitian of the weight update in the real part.

The term $\frac{\partial\|\tilde{e}_a(n)\|_2^2}{\partial\mathbf{w}(n)}$ is equivalent to $\nabla_{\mathbf{w}}E_a(n)$ and is given by

$$\nabla_{\mathbf{w}}E_a = \tilde{e}_a(n)\frac{\partial\tilde{e}_a^*(n)}{\partial\mathbf{w}(n)} + \frac{\partial\tilde{e}_a(n)}{\partial\mathbf{w}(n)}\tilde{e}_a^*(n) = 2\tilde{e}_a(n)\frac{\partial\tilde{e}_a(n)}{\partial\mathbf{w}(n)}. \tag{46}$$

From Appendix, the real part of the product $\mathbf{w}^T(n)\mathbf{x}(n)$ is given as

$$\begin{aligned}\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)_a &= \mathbf{w}_a^T(n)\mathbf{x}_a(n) - \mathbf{w}_b^T(n)\mathbf{x}_b(n)\\ &\quad - \mathbf{w}_c^T(n)\mathbf{x}_c(n) - \mathbf{w}_d^T(n)\mathbf{x}_d(n).\end{aligned} \tag{47}$$

Using (47), the real part of the nonlinear function $\bar{\Phi}_a(\cdot)$ can be expanded into

$$\begin{aligned}\bar{\Phi}_a\big(\mathbf{w}^T(n)\mathbf{x}(n)\big) &= \bar{\Phi}\big(\mathbf{w}_a^T(n)\mathbf{x}_a(n) - \mathbf{w}_b^T(n)\mathbf{x}_b(n)\\ &\quad - \mathbf{w}_c^T(n)\mathbf{x}_c(n) - \mathbf{w}_d^T(n)\mathbf{x}_d(n)\big).\end{aligned} \tag{48}$$

Substitute the expression for the real part of the nonlinear function (48) into the a priori error (43) and then differentiate with respect to $\mathbf{w}(n)$ to give

$$\begin{aligned}\frac{\partial\tilde{e}_a(n)}{\partial\mathbf{w}(n)} &= -\lambda_a(n)\bar{\Phi}_a'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\\ &\quad\times\big(\mathbf{x}_a(n) - \mathbf{x}_b(n)\imath - \mathbf{x}_c(n)\jmath - \mathbf{x}_d(n)\kappa\big)\\ &= -\lambda_a(n)\bar{\Phi}_a'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\mathbf{x}^*(n).\end{aligned} \tag{49}$$

Replacing (49) into the error gradient $\nabla_{\mathbf{w}}E_a(n)$ in (46) gives

$$\nabla_{\mathbf{w}}E_a(n) = -2\tilde{e}_a(n)\lambda_a(n)\bar{\Phi}_a'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\mathbf{x}^*(n). \tag{50}$$

The term $\Delta_a\mathbf{w}^H(n)$ is obtained from (50) and is given by

$$\Delta_a\mathbf{w}^H(n) = \mu\Big(\lambda_a(n)\bar{\Phi}_a'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\mathbf{x}^T(n)\tilde{e}_a(n)\Big). \tag{51}$$

Replace the error gradient $\nabla_{\mathbf{w}}E_a(n)$ from (50) and $\Delta_a\mathbf{w}^H(n)$ from (51) into the real part of the Taylor series expansion (45) to yield

$$\begin{aligned}\|\bar{e}_a(n)\|_2^2 &= \|\tilde{e}_a(n)\|_2^2\\ &\quad - \Big[2\mu\lambda_a^2(n)\mathbf{x}^T(n)\mathbf{x}^*(n)\bar{\Phi}_a'^2\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\|\tilde{e}_a(n)\|_2^2\Big].\end{aligned} \tag{52}$$

Now, in order to satisfy the convergence condition (36), we have

$$0 < 1 - 2\mu\lambda_a^2(n)\mathbf{x}^T(n)\mathbf{x}^*(n)\bar{\Phi}_a'^2\big(\mathbf{w}^T(n)\mathbf{x}(n)\big) < 1. \tag{53}$$

Solving for $\lambda_a(n)$ gives the stability bounds on the adaptive amplitude parameter, in the form

$$0 < \lambda_a^2(n) < \frac{1}{2\mu\mathbf{x}^T(n)\mathbf{x}^*(n)\bar{\Phi}_a'^2\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)} \tag{54}$$

which also reveals the relationship between the value of the amplitude of the quaternion nonlinearity and the stepsize parameter.

Similarly, using the same procedures, the bounds on $\lambda_b(n)$, $\lambda_c(n)$ and $\lambda_d(n)$ can be found as

$$0 < \lambda_b^2(n) < \frac{1}{2\mu\mathbf{x}^T(n)\mathbf{x}^*(n)\bar{\Phi}_b'^2\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)} \tag{55}$$

$$0 < \lambda_c^2(n) < \frac{1}{2\mu\mathbf{x}^T(n)\mathbf{x}^*(n)\bar{\Phi}_c'^2\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)} \tag{56}$$

$$0 < \lambda_d^2(n) < \frac{1}{2\mu\mathbf{x}^T(n)\mathbf{x}^*(n)\bar{\Phi}_d'^2\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)}. \tag{57}$$

5. Simulations

Simulations were performed in an M-step prediction setting and provide a comprehensive comparison between the nonlinear FIR filters trained with SQAFA, AASQAFA, QMLP-FIR, Complex Nonlinear Gradient Descent (CNGD) (Mandic & Goh, 2009), Nonlinear Gradient Descent (NGD) (Mandic & Goh, 2009) and the training algorithm for the Quaternion- valued Multilayer Perceptron (QMLP) (Arena et al., 1998). The SQAFA, AASQAFA, QMLP-FIR, CNGD and NGD were implemented with a filter length L whereas the QMLP had one hidden layer comprising L inputs, three hidden neurons and one output neuron. The nonlinear function was the tanh function applied component-wise. The original QMLP applied the unipolar logistic function as the nonlinearity, whereas the QMLP algorithm implemented in our simulations applied the bipolar tanh function, which was better suited to the dynamic range of the data. Interchanging the nonlinearity would not lead to a significant deviation in performance (Duch & Jankowski, 1999). In the experiments, the amplitudes of input signals in each dimension were scaled to within the range $[-0.8, 0.8]$. The step size of the adaptive amplitude was chosen to be $\rho = 0.4$ with an initial amplitude $\lambda(0) = 1$ for all experiments. A total of 20 independent simulation trials were conducted and averaged.

The standard prediction gain R_p was used as a quantitative measure of performance defined as (Haykin & Li, 1995)

$$R_p = 10\log_{10}\frac{\sigma_x^2}{\sigma_e^2} \tag{58}$$

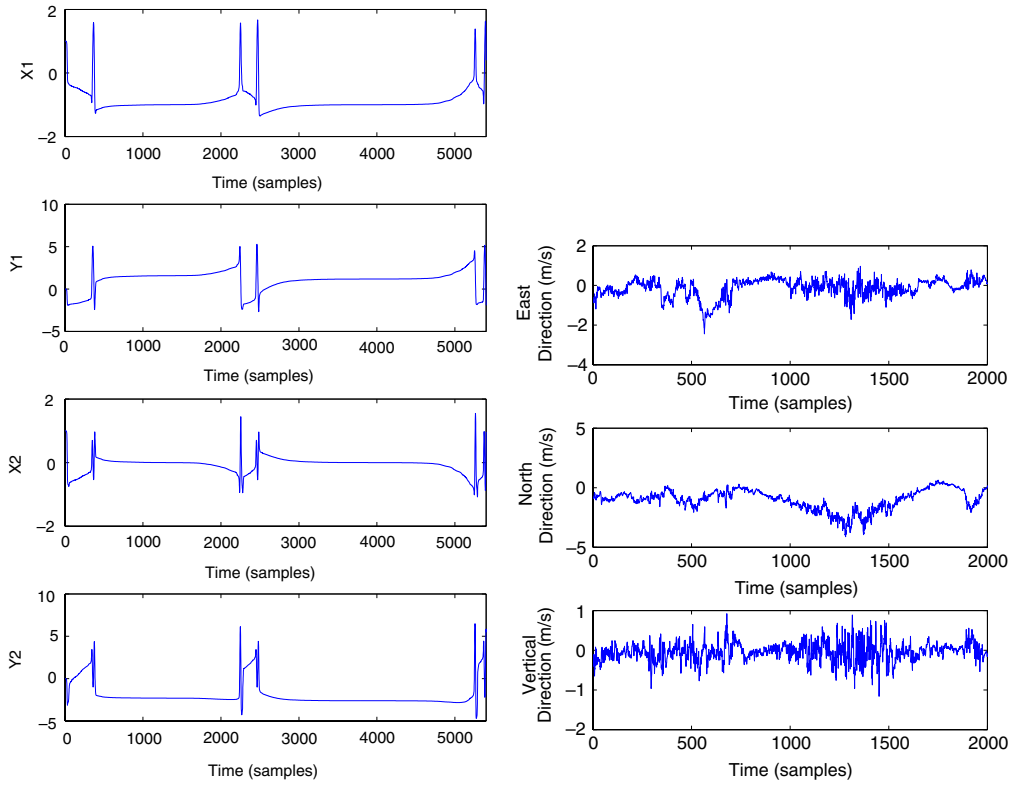
where σ_x^2 and σ_e^2 denote the estimated variance of the input and error, respectively. The variances were estimated according to

$$\sigma_x^2 = E\{x_a^2 + x_b^2 + x_c^2 + x_d^2\} \tag{59}$$

$$\sigma_e^2 = E\{e_a^2 + e_b^2 + e_c^2 + e_d^2\} \tag{60}$$

where $E\{\cdot\}$ denotes the statistical expectation operator, x_a^2, x_b^2, x_c^2, x_d^2 are the corresponding squared components of the input signal, and similarly the squared error components, e_a^2, e_b^2, e_c^2, e_d^2. All these values were measured at the steady-state.

Two quaternion-valued processes were considered: the synthetic benchmark 4D Saito's Chaotic Signal (Mitsubori & Saito, 1994) and the real-world 3D wind field (pure quaternion).



(a) 4D Saito signal. (b) Wind signal.

Fig. 2. Left: The 4D Saito signal. Right: The 3D wind signal.

5.1. Four-dimensional Saito's chaotic circuit

The Saito's chaotic circuit is governed by four state variables and five parameters, and is given by (Mitsubori & Saito, 1994)

$$\begin{bmatrix} \frac{\partial x_1}{\partial \tau} \\ \frac{\partial y_1}{\partial \tau} \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ -\alpha_1 & -\alpha_1\beta_1 \end{bmatrix} \begin{bmatrix} x_1 - \eta\rho_1 h(z) \\ y_1 - \eta\dfrac{\rho_1}{\beta_1} h(z) \end{bmatrix} \tag{61}$$

$$\begin{bmatrix} \frac{\partial x_2}{\partial \tau} \\ \frac{\partial y_2}{\partial \tau} \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ -\alpha_2 & -\alpha_2\beta_2 \end{bmatrix} \begin{bmatrix} x_2 - \eta\rho_2 h(z) \\ y_2 - \eta\dfrac{\rho_2}{\beta_2} h(z) \end{bmatrix} \tag{62}$$

where τ is the time constant of the chaotic circuit and $h(z)$ is the normalized hysteresis value which is given as (Mitsubori & Saito, 1994)

$$h(z) = \begin{cases} 1, & z \geq -1 \\ -1, & z \leq 1. \end{cases} \tag{63}$$

The variables z, ρ_1 and ρ_2 are given as

$$z = x_1 + x_2 \tag{64}$$

$$\rho_1 = \frac{\beta_1}{1 - \beta_1} \tag{65}$$

$$\rho_2 = \frac{\beta_2}{1 - \beta_2}. \tag{66}$$

Saito's chaotic signal used is initialized with the following standard parameters: $\eta = 1.3$, $\alpha_1 = 7.5$, $\alpha_2 = 15$, $\beta_1 = 0.16$ and $\beta_2 = 0.097$. Fig. 2(a) shows the 4D Saito's signal dimension-wise. Fig. 3 illustrates the performance of the algorithms considered as a function of the prediction horizon M (with $\mu = 10^{-2}$), and as a function of stepsize μ (with the prediction horizon, $M = 1$). From Fig. 3, it can be seen that AASQAFA and SQAFA have similar performance and they both outperform QMLP.

5.2. Wind forecasting

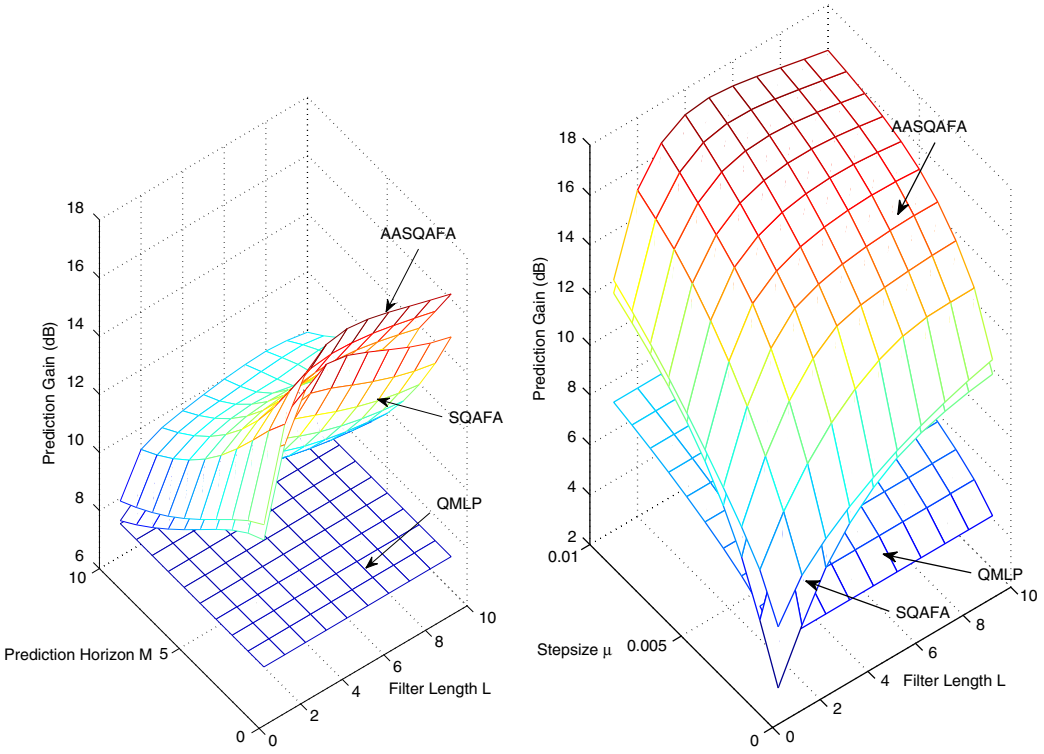
In the next simulation, a 3D wind field was used as an input.[2] The wind data were initially sampled at 50 Hz, but resampled at 5 Hz for simulation purposes. Fig. 2(b) shows the 3D wind data dimension-wise. Fig. 4 depicts the performance of SQAFA, AASQAFA and QMLP as a function of the prediction horizon M and stepsize μ. The prediction gain for SQAFA was better than that of QMLP in both case studies (varying learning rate and prediction horizon), thus indicating the benefits of fully exploiting the quaternion algebra. The performance of AASQAFA was superior to that of SQAFA, due to its adaptive amplitude which follows the dynamics of the wind signal more closely.

Fig. 5 shows the comparison between SQAFA, the learning algorithm for QMLP applied to the FIR filter (QFIR), CNGD, and NGD as a function of prediction horizon M and stepsize μ. The performance gain for SQAFA was considerably higher than that for QFIR, followed by those of the NGD algorithm and CNGD algorithm. When using the same FIR architecture, the SQAFA outperformed the QFIR, thus highlighting the advantage of exploiting the non-commutativity aspect of quaternion algebra. Moreover, both quaternion based algorithms proved superior to their complex and real counterparts.

6. Discussion

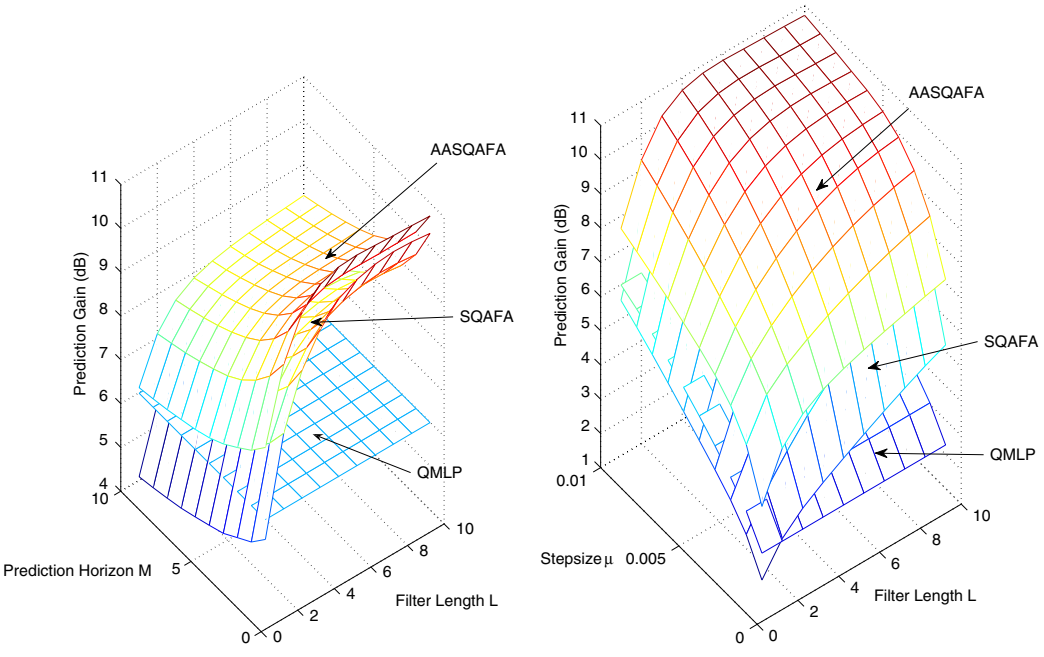
The performance of the SQAFA was generally better than that of QMLP, as it takes into account more complete information about

[2] The wind data are obtained from Prof. K. Aihara and his team at the Institute for Industrial Science, University of Tokyo, in an urban environment.



(a) Dependence of prediction gain on M and L. (b) Dependence of prediction gain on μ and L.

Fig. 3. Performance of SQAFA, AASQAFA and QMLP on the prediction of 4D Saito's chaotic signal.

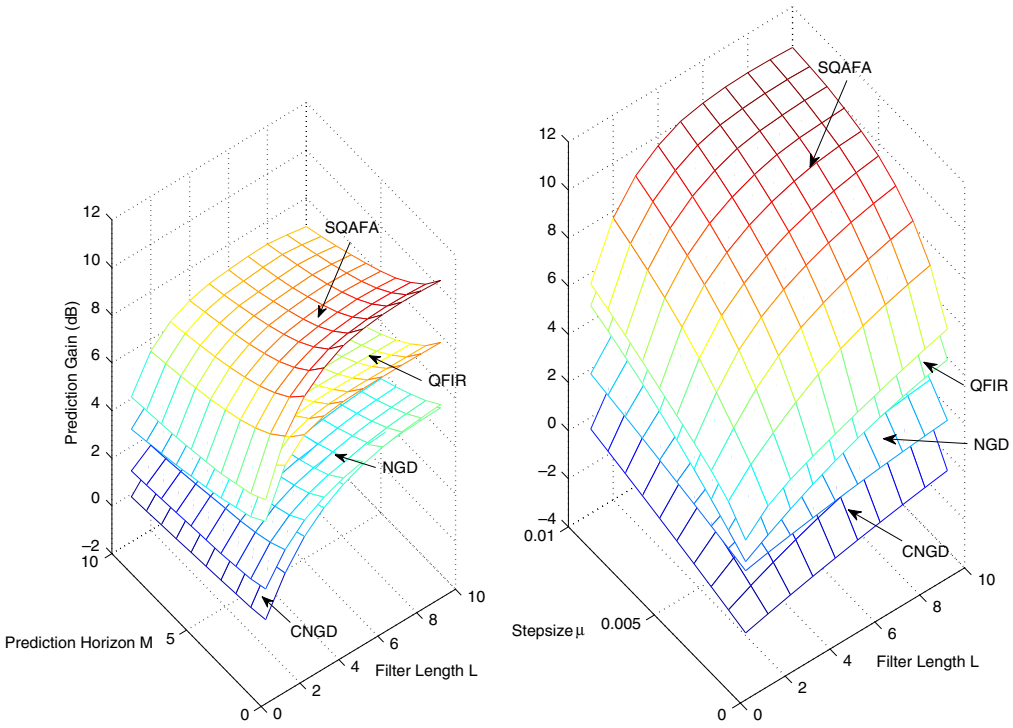


(a) Dependence of prediction gain on M and L. (b) Dependence of prediction gain on μ and L.

Fig. 4. Performance of SQAFA, AASQAFA and QMLP on the prediction of 3D wind signal.

the statistics of the multidimensional signal. The AASQAFA, on the other hand, outperformed SQAFA due to its ability to better track the dynamics of the signal. The QMLP was less affected by the length of the prediction horizon and the filter length as compared with the SQAFA and AASQAFA. The deterioration of the QMLP prediction gain with the increase of prediction horizon is almost negligible due to the structural richness of the multilayer neural network compared to the single layer FIR architecture of SQAFA and AASQAFA. Both the $\mathbb{H}$ domain algorithms outperformed the algorithms in the $\mathbb{C}$ and $\mathbb{R}$ indicating quaternion based signal processing being a better choice for the processing of three-dimensional and hypercomplex processes.

Another aspect that needs to be addressed is the computational complexity of the algorithms, which is summarized in Table 1. The computational complexities for AASQAFA and SQAFA are both $\mathcal{O}(68L)$ and QMLP is $\mathcal{O}(108L)$. On the other hand, the



(a) Dependence of prediction gain on M and L.

(b) Dependence of prediction gain on μ and L.

Fig. 5. Performance of SQAFA, QFIR, CNGD and NGD on the prediction of 3D wind signal.

Table 1
Computational complexities of the algorithms.

Algorithms	Additions	Multiplications
$1 \times$ QMLP	$96L + 168$	$108L + 216$
$1 \times$ SQAFA	$60L + 18$	$68L + 24$
$1 \times$ AASQAFA	$60L + 22$	$68L + 36$
$1 \times$ QMLP-FIR	$32L + 18$	$36L + 24$
$2 \times$ CNGD	$16L + 4$	$24L + 8$
$4 \times$ NGD	$8L + 4$	$12L + 4$

computational complexity of QMLP is more than twice that of SQAFA and AASQAFA when $L = 1$. Since the computational complexities for the SQAFA and AASQAFA are similar, AASQAFA is a preferable choice due to its superior performance. Computational complexities of the QFIR is $\mathcal{O}(36L)$, for the CNGD it is $\mathcal{O}(24L)$, and for NGD it is $\mathcal{O}(12L)$. The computational complexity of SQAFA and AASQAFA are less than two times that of QFIR, nearly three times that of CNGD and almost seven times that of NGD. Hence, there is a trade off between a higher computational complexity and increment in performance.

The QMLPs were proven to be universal approximators in Arena et al. (1998). Specifically, it was shown that a universal approximator for quaternion functions must be in the form of (Arena et al., 1998)

$$g(x) = \sum_{i=1}^{N} C_i \Phi\left(\mathbf{w}^T(n)\mathbf{x}(n) + \theta(n)\right) \tag{67}$$

where $g(x)$ is a quaternion-valued function to be approximated, C_i is quaternion-valued variable, $\Phi(\cdot)$ is a split quaternion sigmoidal function, $\mathbf{w}(n)$ is the quaternion weight vectors, $\mathbf{x}(n)$ is the quaternion input vectors and $\theta(n)$ is the quaternion-valued bias term. Eq. (67) conforms with the earlier findings of Cybenko (1989), who stated that any continuous function can be approximated by the superposition of N sigmoidal functions. In the context of the SQAFA and AASQAFA, $N = 1$, and, therefore, if SQAFA and AASQAFA are extended to a neural network architecture, their approximation capabilities become those of a universal approximator.

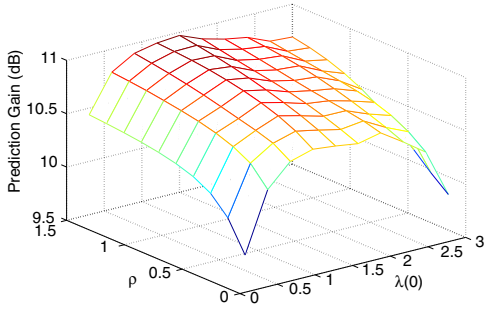


Fig. 6. Prediction gain of AASQAFA for the varying initial amplitude $\lambda(0)$ and step size ρ.

Fig. 6 illustrates the dependence of the prediction gain of AASQAFA on the initial amplitude $\lambda(0)$ and step size ρ. It is shown that AASQAFA is robust to the initial state $\lambda(0)$ and the learning rate ρ for the realistic range of $0 < \lambda(0) < 3$ and $0.1 \leq \rho \leq 1.5$.

In summary, the advantages SQAFA and AASQAFA are

- Taking into account the non-commutativity of the quaternion product leads to more efficient use of the available statistics and improved performance;
- AASQAFA caters for the changes in dynamical range of the signals, resulting in a performance enhancement;
- AASQAFA is robust to the choice of initial amplitude $\lambda(0)$ and learning rate ρ.

7. Conclusion

A class of stochastic gradient algorithms (SQAFA and AASQAFA) for the training of quaternion-valued nonlinear adaptive FIR filters has been proposed. The learning algorithm for the training of QMLP

proved inadequate when modelling the hypercomplex processes considered (four-dimensional Saito's chaotic signal and three-dimensional wind signal) due to the strong coupling between each dimension. Furthermore, multiple univariate NGD and a pair of complex NGD (CNGD) were also considered, but yielded poorer performance when compared with both the QMLP and the SQAFA algorithms. The "split quaternion" nonlinear function was next employed, as there are no analytic extensions of elementary transcendental functions from $\mathbb{C}$ to $\mathbb{H}$, due to the violation of the Cauchy–Riemann–Fueter condition.

The derivations of the SQAFA and AASQAFA have taken into account the non-commutativity of the quaternion product, and have been simplified by making use of the odd-symmetry of elementary transcendental functions applied component-wise. A rigorous stability analysis has provided the range of the stepsizes for SQAFA and AASQAFA, and has established the relationship between the adaptive amplitude and the stepsize of the AASQAFA. The proposed algorithms (SQAFA and AASQAFA) have been shown to exhibit excellent performance on the prediction of quaternion-valued real-world vector fields. The AASQAFA achieved better performance due to its enhanced ability to track the time varying dynamics of the input signals.

Appendix. Derivation of $\nabla_{\mathbf{w}} y(n)$

To calculate $\nabla_{\mathbf{w}} y(n)$ and $\nabla_{\mathbf{w}} y^*(n)$, terms $\mathbf{w}^T(n)\mathbf{x}(n)$ and $\mathbf{x}^H(n)\mathbf{w}^*(n)$ are first expanded as (due to space limitation, the time index "n" has been dropped) :

$$\mathbf{w}^T(n)\mathbf{x}(n) = \begin{bmatrix} \mathbf{w}_a^T\mathbf{x}_a - \mathbf{w}_b^T\mathbf{x}_b - \mathbf{w}_c^T\mathbf{x}_c - \mathbf{w}_d^T\mathbf{x}_d \\ \mathbf{w}_a^T\mathbf{x}_b + \mathbf{w}_b^T\mathbf{x}_a + \mathbf{w}_c^T\mathbf{x}_d - \mathbf{w}_d^T\mathbf{x}_c \\ \mathbf{w}_a^T\mathbf{x}_c + \mathbf{w}_c^T\mathbf{x}_a + \mathbf{w}_d^T\mathbf{x}_b - \mathbf{w}_b^T\mathbf{x}_d \\ \mathbf{w}_a^T\mathbf{x}_d + \mathbf{w}_d^T\mathbf{x}_a + \mathbf{w}_b^T\mathbf{x}_c - \mathbf{w}_c^T\mathbf{x}_b \end{bmatrix} \tag{A.1}$$

$$\mathbf{x}^H(n)\mathbf{w}^*(n) = \begin{bmatrix} \mathbf{w}_a^T\mathbf{x}_a - \mathbf{w}_b^T\mathbf{x}_b - \mathbf{w}_c^T\mathbf{x}_c - \mathbf{w}_d^T\mathbf{x}_d \\ -\mathbf{w}_a^T\mathbf{x}_b - \mathbf{w}_b^T\mathbf{x}_a - \mathbf{w}_c^T\mathbf{x}_d + \mathbf{w}_d^T\mathbf{x}_c \\ -\mathbf{w}_a^T\mathbf{x}_c - \mathbf{w}_c^T\mathbf{x}_a - \mathbf{w}_d^T\mathbf{x}_b + \mathbf{w}_b^T\mathbf{x}_d \\ -\mathbf{w}_a^T\mathbf{x}_d - \mathbf{w}_d^T\mathbf{x}_a - \mathbf{w}_b^T\mathbf{x}_c + \mathbf{w}_c^T\mathbf{x}_b \end{bmatrix} \tag{A.2}$$

and the gradients $\nabla_{\mathbf{w}} y(n)$ and $\nabla_{\mathbf{w}} y^*(n)$ are defined as:

$$\nabla_{\mathbf{w}} y(n) = \nabla_{\mathbf{w}_a} y(n) + \nabla_{\mathbf{w}_b} y(n)\imath + \nabla_{\mathbf{w}_c} y(n)\jmath + \nabla_{\mathbf{w}_d} y(n)\kappa \tag{A.3}$$

$$\nabla_{\mathbf{w}} y^*(n) = \nabla_{\mathbf{w}_a} y^*(n) + \nabla_{\mathbf{w}_b} y^*(n)\imath + \nabla_{\mathbf{w}_c} y^*(n)\jmath + \nabla_{\mathbf{w}_d} y^*(n)\kappa. \tag{A.4}$$

Using the odd-symmetry property of the elementary transcendental functions as we have:

$$\begin{aligned} \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big) &= \Phi_a'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big) - \Phi_b'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\imath \\ &\quad - \Phi_c'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\jmath - \Phi_d'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\kappa \\ &= \Phi'\big(\mathbf{x}^H(n)\mathbf{w}(n)\big). \end{aligned} \tag{A.5}$$

Based on the expansions (A.1) and (A.2), the derivatives of (A.3) can be computed as:

$$\begin{aligned} \nabla_{\mathbf{w}_a} y(n) &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(\mathbf{x}_a + \mathbf{x}_b\imath + \mathbf{x}_c\jmath + \mathbf{x}_d\kappa) \\ \nabla_{\mathbf{w}_b} y(n)\imath &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_b + \mathbf{x}_a\imath - \mathbf{x}_d\jmath + \mathbf{x}_c\kappa)\imath \\ &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_a - \mathbf{x}_b\imath + \mathbf{x}_c\jmath + \mathbf{x}_d\kappa) \\ \nabla_{\mathbf{w}_c} y(n)\jmath &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_c + \mathbf{x}_d\imath + \mathbf{x}_a\jmath - \mathbf{x}_b\kappa)\jmath \\ &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_a + \mathbf{x}_b\imath - \mathbf{x}_c\jmath + \mathbf{x}_d\kappa) \\ \nabla_{\mathbf{w}_d} y(n)\kappa &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_d - \mathbf{x}_c\imath + \mathbf{x}_b\jmath + \mathbf{x}_a\kappa)\kappa \\ &= \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_a + \mathbf{x}_b\imath + \mathbf{x}_c\jmath - \mathbf{x}_d\kappa) \quad . \end{aligned} \tag{A.6}$$

Similar to the above and using the odd-symmetry property (A.5), the derivatives in (A.4) are obtained as

$$\begin{aligned} \nabla_{\mathbf{w}_a} y^*(n) &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(\mathbf{x}_a - \mathbf{x}_b\imath - \mathbf{x}_c\jmath - \mathbf{x}_d\kappa) \\ \nabla_{\mathbf{w}_b} y^*(n)\imath &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_b - \mathbf{x}_a\imath + \mathbf{x}_d\jmath - \mathbf{x}_c\kappa)\imath \\ &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(\mathbf{x}_a - \mathbf{x}_b\imath - \mathbf{x}_c\jmath - \mathbf{x}_d\kappa) \\ \nabla_{\mathbf{w}_c} y^*(n)\jmath &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_c - \mathbf{x}_d\imath - \mathbf{x}_a\jmath + \mathbf{x}_b\kappa)\jmath \\ &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(\mathbf{x}_a - \mathbf{x}_b\imath - \mathbf{x}_c\jmath - \mathbf{x}_d\kappa) \\ \nabla_{\mathbf{w}_d} y^*(n)\kappa &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(-\mathbf{x}_d + \mathbf{x}_c\imath - \mathbf{x}_b\jmath - \mathbf{x}_a\kappa)\kappa \\ &= \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)(\mathbf{x}_a - \mathbf{x}_b\imath - \mathbf{x}_c\jmath - \mathbf{x}_d\kappa) \end{aligned} \tag{A.7}$$

Substituting (A.6) into (A.3) yields:

$$\nabla_{\mathbf{w}} y(n) = \Phi'\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\big(-2\mathbf{x}^*(n)\big) \tag{A.8}$$

and substituting (A.7) into (A.4) gives:

$$\nabla_{\mathbf{w}} y^*(n) = \Phi'^*\big(\mathbf{w}^T(n)\mathbf{x}(n)\big)\big(4\mathbf{x}^*(n)\big) \tag{A.9}$$

which is employed in the derivation of the SQAFA and AASQAFA.

References

Arena, P., Fortuna, L., Muscato, G., & Xibilia, M. G. (1998). *Lecture notes in control and information sciences*: *Vol. 234. Neural networks in multidimensional domains*. Springer Verlag.

Biamino, D., Cannata, G., Maggiali, M., & Piazza, A. (2005). MAC-EYE: A tendon driven fully embedded robot eye. In *Proceedings IEEE-RAS international conference on humanoid robots* (pp. 62–67).

Buchholz, S., & Bihan, N. L. (2008). Polarized signal classification by complex and quaternionic multi-layer perceptrons. *International Journal of Neural Systems*, *18*(2), 75–85.

Cheong-Took, C., & Mandic, D. P. (2009). The quaternion LMS algorithm for adaptive filtering of hypercomplex processes. *IEEE Transactions on Signal Processing*, *57*(4), 1316–1327.

Choe, S. B., & Faraway, J. J. (2004). Modeling head and hand orientation during motion using quaternions. *Journal of Aerospace*, *113*, 186–192.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics Control of Signal, and Systems*, *2*, 303–314.

Duch, W., & Jankowski, N. (1999). Survey of neural transfer functions. *Neural Computing Surveys*, *2*, 163–212.

Georgiou, G. M., & Koutsougeras, C. (1992). Complex domain Backpropagation. *IEEE Transactions on Circuits and Systems II*, *39*(5), 330–334.

Goh, S. L., & Mandic, D. P. (2003). Recurrent neural networks with trainable amplitude of activation functions. *Neural Networks*, *16*(8), 1095–1100.

Hanna, A. I., & Mandic, D. P. (2002). Nonlinear FIR adaptive filters with a gradient adaptive amplitude in the nonlinearity. *IEEE Signal Processing Letters*, *9*(8), 253–255.

Haykin, S. (2002). *Adaptive filter theory* (4th ed.). Prentice Hall.

Haykin, S., & Li, L. (1995). Nonlinear adaptive prediction of nonstationary signals. *IEEE Transactions on Signal Processing*, *43*(2), 526–535.

Heaviside, O. (1893). Vectors versus quaternions. *Nature*, *47*, 533–534.

Karney, C. F. F. (2007). Quaternions in molecular modelling. *Journal of Molecular Graphics and Modelling*, *25*(5), 595–604.

Kim, T., & Adali, T. (2003). Approximation by fully complex multilayer perceptrons. *Neural Computation*, *15*(7), 1641–1666.

Leung, H., & Haykin, S. (1991). The complex Backpropagation algorithm. *IEEE Transactions on Signal Processing*, *39*(9), 2101–2104.

MacFarlane, A. (1893). Vectors versus quaternions. *Nature*, *48*, 75–76.

Mandic, D. P., & Chambers, J. A. (1999). Relating the slope of the activation function and the learning rate within a recurrent neural network. *Neural Computation*, *11*(5), 1069–1077.

Mandic, D. P., & Chambers, J. A. (2001). *Recurrent neural networks for prediction: Learning algorithms, architectures and stability*. Wiley.

Mandic, D. P., & Goh, S. L. (2009). *Complex valued nonlinear adaptive filters: Noncircularity, widely linear and neural models*. Wiley.

Mitsubori, K., & Saito, T. (1994). Torus doubling and hyperchaos in a five dimensional hysteresis circuit. In *Proceedings of 1994 IEEE international symposium on circuit and systems* (pp. 113–116). *Vol. 6*.

Nitta, T. (1993). A back-propagation algorithm for neural networks based on 3D vector product. In *Proceedings of international joint conference on neural networks* (pp. 589–592). *Vol. 1*.

Nitta, T., & de Garis, H. (1992). A 3D vector version of the back-propagation algorithm. In *Proceedings of international joint conference on neural networks* (pp. 511–516). *Vol. 2*.

Silva, C. C., & Martins, R. D. A. (2002). Polar and axial vectors versus quaternions. *American Association of Physics Teachers*, *70*(9), 958–963.

Soria-Olivas, E., Maravilla, J., Guerrero-Martinez, J. F., Martinez-Sober, M., & Espi-Lopez, J. (1998). An easy demonstration of the optimum value of the adaptation constant in the LMS algorithm. *IEEE Transactions on Education*, *41*(1), 81.

Sudbery, A. (1979). Quaternionic analysis. *Mathematical Proceedings of the Cambridge Philosophical Society*, *85*(2), 199–225.

Trentin, E. (2001). Networks with trainable amplitude of activation functions. *Neural Networks*, *14*(4–5), 471–493.