Optimization in Quaternion Dynamic Systems: Gradient, Hessian, and Learning Algorithms

Dongpo Xu, Yili Xia, *Member, IEEE*, and Danilo P. Mandic, *Fellow, IEEE*

***Abstract*— The optimization of real scalar functions of quaternion variables, such as the mean square error or array output power, underpins many practical applications. Solutions typically require the calculation of the gradient and Hessian. However, real functions of quaternion variables are essentially nonanalytic, which are prohibitive to the development of quaternion-valued learning systems. To address this issue, we propose new definitions of quaternion gradient and Hessian, based on the novel generalized Hamilton-real (GHR) calculus, thus making a possible efficient derivation of general optimization algorithms directly in the quaternion field, rather than using the isomorphism with the real domain, as is current practice. In addition, unlike the existing quaternion gradients, the GHR calculus allows for the product and chain rule, and for a one-to-one correspondence of the novel quaternion gradient and Hessian with their real counterparts. Properties of the quaternion gradient and Hessian relevant to numerical applications are also introduced, opening a new avenue of research in quaternion optimization and greatly simplified the derivations of learning algorithms. The proposed GHR calculus is shown to yield the same generic algorithm forms as the corresponding real- and complex-valued algorithms. Advantages of the proposed framework are illuminated over illustrative simulations in quaternion signal processing and neural networks.**

***Index Terms*— Backpropagation, generalized Hamilton-real (GHR) calculus, nonlinear adaptive filtering, quaternion gradient, quaternion least mean square (QLMS), quaternion optimization, real-time recurrent learning.**

I. Introduction

QUATERNION algebra has recently attracted a considerable research interest in areas including color image processing [1], [2], automatic control [3], aerospace and satellite tracking [4], bearings-only tracking [5], body motion tracking [6]–[8], land classification [9], modeling of wind profile [10], [11], and the processing of polarized waves [12], [13]. In these areas, quaternions have allowed for a reduction in the number of parameters and operations involved compared with vector algebra, and for physically meaningful representation. Despite the obvious advantages, reflected in the fact that quaternions have become a standard in computer graphics [14], the main obstacle for their more widespread use in dynamic systems has been the nonanalytic nature of the real-valued cost (objective) functions of quaternion variables, which naturally arise in optimization procedures [15]–[17]. Currently used pseudoderivatives do circumvent this issue for small-scale problems by rewriting a real cost function $J(q)$ in terms of the four real components of the quaternion variable, and then taking separately the real derivatives with respect to these independent real parts. In this way, $J(q)$ is treated as a real analytic mapping between $\mathbb{R}^4$ and $\mathbb{R}$. However, since the original framework is quaternion valued, it is often awkward to reformulate the problem in the real domain and very tedious to calculate gradients for the optimization in even moderately complex quaternion dynamic systems.

To this end, the recent Hamilton-real (HR) calculus [18] has made it possible to take the formal derivatives of the real $J(q)$ with respect to the quaternion variable and its involutions. This approach saves on computational burden and greatly simplifies gradient expressions, it can also be considered as a generalization of the complex-real (CR) calculus [19]–[21] to the quaternion field, as the basis for the HR calculus is the use of involutions (generalized conjugate) [22]. However, the traditional product rule does not apply within the HR calculus because of the noncommutativity of quaternion algebra. To this end, we introduced the generalized HR (GHR) calculus in order to equip quaternion analysis with both the product and chain rules [23], leading to the integral form of the mean value theorem and Taylor's theorem. This makes it possible, for the first time, to directly apply the optimization techniques and learning algorithms in the quaternion field, rather than transforming the problem to the real domain. This also yields concise and elegant algorithm forms and opens new opportunities in neural and dynamic systems.

We here further extend the concept of GHR calculus to rigorously define quaternion gradient and Hessian. This promises similar advantages to those enabled by the CR calculus in the complex domain [21], [24], [29]—the complex gradient and Hessian derived by the CR calculus were instrumental for the developments in complex-domain

Manuscript received August 14, 2014; revised March 3, 2015 and May 12, 2015; accepted May 31, 2015. Date of publication June 16, 2015; date of current version January 18, 2016. This work was supported in part by the Natural Science Foundation of Jiangsu Province under Grant BK20140645, in part by the Research Fund for the Doctoral Program of Higher Education of China under Grant 20122304120028, and in part by the National Natural Science Foundation of China under Grant 61301202 and Grant 61401094.

D. Xu is with the School of Mathematics and Statistics, Northeast Normal University, Changchun 130024, China, also with the College of Science, Harbin Engineering University, Harbin 150001, China, and also with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: dongpoxu@gmail.com).

Y. Xia is with the School of Information Science and Engineering, Southeast University, Nanjing 210096, China, and also with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: yili.xia06@gmail.com).

D. P. Mandic is with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: d.mandic@imperial.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2015.2440473

optimization [25] and signal processing [26]–[29]. Next, the basic relationships between the quaternion gradient and Hessian and their real counterparts are established by invertible linear transforms, these are shown to be very convenient for the derivation of algorithms based on first- and second-order Taylor series expansion (TSE) in the quaternion field. We also illuminate that the quadrivariate versus quaternion relations involve redundancy, since they operate in the augmented quaternion space $\mathbb{H}^{4N\times 1}$, which motivates us to propose an efficient way to obtain the algorithms that operate directly in $\mathbb{H}^{N\times 1}$. This paper concludes with the examples of several applications in learning systems enabled by the GHR framework, such as the quaternion least mean square (QLMS), nonlinear adaptive filtering, and in neurodynamic systems (quaternion-valued feedforward and recurrent neural networks).

II. Preliminaries

A. Quaternion Algebra

In 1843, Sir W. R. Hamilton invented the quaternion algebra, denoted $\mathbb{H}$ in his honor. Quaternions are an associative but not commutative algebra over $\mathbb{R}$, defined as[1]

$$\mathbb{H} \triangleq \{q_a + iq_b + jq_c + kq_d \mid q_a, q_b, q_c, q_d \in \mathbb{R}\} \tag{1}$$

where $\{1, i, j, k\}$ is a basis of $\mathbb{H}$, and the imaginary units i, j, and k satisfy $i^2 = j^2 = k^2 = ijk = -1$, which implies $ij = k = -ji$, $jk = i = -kj$, and $ki = j = -ik$. A distinguishing feature is that the multiplication of two quaternions is noncommutative, for example, $ij \neq ji = -k$. For any quaternion

$$q = q_a + iq_b + jq_c + kq_d = S_q + V_q \tag{2}$$

the real (scalar) part is denoted by $q_a = S_q = \Re(q)$, whereas the vector part (also called *pure* quaternion) $V_q = \Im(q) = iq_b + jq_c + kq_d$ comprises the three imaginary parts. The real part $S_q = q_a$ behaves like a scalar in $\mathbb{R}$, and the vector part $V_q = iq_b + jq_c + kq_d$ behaves like a vector $\vec{v} = (q_b, q_c, q_d)$ in a 3-D vector space. The conjugate of a quaternion q is $q^* = q_a - iq_b - jq_c - kq_d$, while the conjugate of the product satisfies $(pq)^* = q^*p^*$. The modulus of a quaternion is defined as $|q| = \sqrt{qq^*}$, it is easy to show that $|pq| = |p||q|$. The inverse of a quaternion $q \neq 0$ is $q^{-1} = q^*/|q|^2$, and an important property of the inverse is $(pq)^{-1} = q^{-1}p^{-1}$. If $|q| = 1$, we call q a *unit* quaternion. If $\Re(q) = 0$, then $q^* = -q$ and $q^2 = -|q|^2$. Thus, a *pure unit* quaternion is a square root of -1, such as the imaginary units i, j, and k.

Quaternions can also be written in the *polar* form $q = |q|(\cos\theta + \hat{q}\sin\theta)$, where $\hat{q} = V_q/|V_q|$ is a pure unit quaternion and $\theta = \arccos(\Re(q)/|q|) \in \mathbb{R}$ is the angle (or argument) of the quaternion. We shall next introduce the quaternion rotation and involution operations.

Definition 1 (Quaternion Rotation [30, p. 81]): For any quaternion q, the transformation

$$q^\mu \triangleq \mu q \mu^{-1} \tag{3}$$

[1]For advanced reading on quaternions, we refer to [30], and to [31] for results on matrices of quaternions.

geometrically describes a 3-D rotation of the vector part of q by an angle 2θ about the vector part of μ, where $\mu = |\mu|(\cos\theta + \hat{\mu}\sin\theta)$ is any nonzero quaternion.

In particular, if μ in (3) is a pure unit quaternion, then the quaternion rotation (3) becomes quaternion involution [22], such as q^i, q^j, and q^k, defined by

$$\begin{aligned} q^i &= -iqi = q_a + iq_b - jq_c - kq_d \\ q^j &= -jqj = q_a - iq_b + jq_c - kq_d \\ q^k &= -kqk = q_a - iq_b - jq_c + kq_d. \end{aligned} \tag{4}$$

The properties of the quaternion rotation that are important for this paper (the proof of [32, eqs. 5 and 6]) are

$$q^\mu = q^{\left(\frac{\mu}{|\mu|}\right)}, \quad (pq)^\mu = p^\mu q^\mu, \quad pq = q^p p = qp^{(q^*)} \quad \forall p, q \in \mathbb{H} \tag{5}$$

$$q^{\mu\nu} = (q^\nu)^\mu, \quad q^{\mu *} \triangleq (q^*)^\mu = (q^\mu)^* \triangleq q^{*\mu} \quad \forall \nu, \mu \in \mathbb{H} \tag{6}$$

where $\mu/|\mu|$ is an unit quaternion, that is, $|\mu/|\mu|| = 1$. Therefore, the quaternion μ in (3)–(6) is not required to be a unit quaternion, as $q^\mu = q^{(\mu/|\mu|)}$. Note that the real representation in (1) can be easily generalized to a general orthogonal system $\{1, i^\mu, j^\mu, k^\mu\}$, where the following properties hold [23], [30]:

$$i^\mu i^\mu = j^\mu j^\mu = k^\mu k^\mu = i^\mu j^\mu k^\mu = -1. \tag{7}$$

B. GHR Calculus

Standard quaternion pseudoderivatives represent component-wise real derivatives of quaternion univariate components and are thus a very restrictive tool for the development of learning algorithms, due to their cumbersome and tedious computations [10]. A recent, more elegant, approach that is also applied to cost functions, is the HR calculus, whereby the HR derivatives are given by [18]

$$\begin{pmatrix} \frac{\partial f}{\partial q} \\ \frac{\partial f}{\partial q^i} \\ \frac{\partial f}{\partial q^j} \\ \frac{\partial f}{\partial q^k} \end{pmatrix} = \frac{1}{4}\begin{pmatrix} 1 & -i & -j & -k \\ 1 & -i & j & k \\ 1 & i & -j & k \\ 1 & i & j & -k \end{pmatrix} \begin{pmatrix} \frac{\partial f}{\partial q_a} \\ \frac{\partial f}{\partial q_b} \\ \frac{\partial f}{\partial q_c} \\ \frac{\partial f}{\partial q_d} \end{pmatrix} \tag{8}$$

and the conjugate HR derivatives (HR* derivatives)

$$\begin{pmatrix} \frac{\partial f}{\partial q^*} \\ \frac{\partial f}{\partial q^{i*}} \\ \frac{\partial f}{\partial q^{j*}} \\ \frac{\partial f}{\partial q^{k*}} \end{pmatrix} = \frac{1}{4}\begin{pmatrix} 1 & i & j & k \\ 1 & i & -j & -k \\ 1 & -i & j & -k \\ 1 & -i & -j & k \end{pmatrix} \begin{pmatrix} \frac{\partial f}{\partial q_a} \\ \frac{\partial f}{\partial q_b} \\ \frac{\partial f}{\partial q_c} \\ \frac{\partial f}{\partial q_d} \end{pmatrix}. \tag{9}$$

Remark 2: It is important to note that the traditional product and chain rules are not valid for the HR calculus. For example, $f(q) = |q|^2$, then from (8) $(\partial|q|^2/\partial q) = (1/2)q^*$, but $(\partial|q|^2/\partial q) \neq q(\partial q^*/\partial q) + (\partial q/\partial q)q^* = -(1/2)q + q^*$.

This difficulty has been solved within the framework of the GHR calculus, which incorporates the novel product and chain rules (12)–(15), see [23] for more detail.

Definition 3 (GHR Derivatives [23]): Let $f : \mathbb{H} \to \mathbb{H}$. Then, the left GHR derivatives of $f(q)$ with respect to q^μ and $q^{\mu *}$ ($\mu \neq 0, \mu \in \mathbb{H}$) are defined as

$$\frac{\partial f}{\partial q^\mu} = \frac{1}{4}\left(\frac{\partial f}{\partial q_a} - \frac{\partial f}{\partial q_b} i^\mu - \frac{\partial f}{\partial q_c} j^\mu - \frac{\partial f}{\partial q_d} k^\mu\right) \in \mathbb{H}$$
$$\frac{\partial f}{\partial q^{\mu *}} = \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + \frac{\partial f}{\partial q_b} i^\mu + \frac{\partial f}{\partial q_c} j^\mu + \frac{\partial f}{\partial q_d} k^\mu\right) \in \mathbb{H} \quad (10)$$

while the right GHR derivatives are defined as

$$\frac{\partial_r f}{\partial q^\mu} = \frac{1}{4}\left(\frac{\partial f}{\partial q_a} - i^\mu \frac{\partial f}{\partial q_b} - j^\mu \frac{\partial f}{\partial q_c} - k^\mu \frac{\partial f}{\partial q_d}\right) \in \mathbb{H}$$
$$\frac{\partial_r f}{\partial q^{\mu *}} = \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + i^\mu \frac{\partial f}{\partial q_b} + j^\mu \frac{\partial f}{\partial q_c} + k^\mu \frac{\partial f}{\partial q_d}\right) \in \mathbb{H} \quad (11)$$

where $\partial f/\partial q_a, \partial f/\partial q_b, \partial f/\partial q_c$, and $\partial f/\partial q_d \in \mathbb{H}$ are the partial derivatives of f with respect to q_a, q_b, q_c, and q_d, respectively, and the set $\{1, i^\mu, j^\mu, k^\mu\}$ is a general orthogonal basis of $\mathbb{H}$.

Some properties of the left GHR derivatives are [23]

$$\text{Product rule}: \frac{\partial (fg)}{\partial q^\mu} = f\frac{\partial g}{\partial q^\mu} + \frac{\partial (fg)}{\partial q^{g\mu}} g \quad (12)$$

$$\text{Product rule}: \frac{\partial (fg)}{\partial q^{\mu *}} = f\frac{\partial g}{\partial q^{\mu *}} + \frac{\partial (fg)}{\partial q^{g\mu *}} g \quad (13)$$

$$\text{Chain rule}: \frac{\partial f(g(q))}{\partial q^\mu} = \sum_{\nu \in \{1,i,j,k\}} \frac{\partial f}{\partial g^\nu}\frac{\partial g^\nu}{\partial q^\mu} \quad (14)$$

$$\text{Chain rule}: \frac{\partial f(g(q))}{\partial q^{\mu *}} = \sum_{\nu \in \{1,i,j,k\}} \frac{\partial f}{\partial g^{\nu *}}\frac{\partial g^{\nu *}}{\partial q^{\mu *}} \quad (15)$$

$$\text{Rotation rule}: \left(\frac{\partial f}{\partial q^\mu}\right)^\nu = \frac{\partial f^\nu}{\partial q^{\nu\mu}}, \quad \left(\frac{\partial f}{\partial q^{\mu *}}\right)^\nu = \frac{\partial f^\nu}{\partial q^{\nu\mu *}} \quad (16)$$

$$\text{Conjugate rule}: \left(\frac{\partial f}{\partial q^\mu}\right)^* = \frac{\partial_r f^*}{\partial q^{\mu *}}, \quad \left(\frac{\partial f}{\partial q^{\mu *}}\right)^* = \frac{\partial_r f^*}{\partial q^\mu} \quad (17)$$

$$\text{if } f \text{ is real} \left(\frac{\partial f}{\partial q^\mu}\right)^* = \frac{\partial f}{\partial q^{\mu *}}, \quad \left(\frac{\partial f}{\partial q^{\mu *}}\right)^* = \frac{\partial f}{\partial q^\mu}. \quad (18)$$

Example 4: Find the GHR derivatives of the functions

$$f(q) = \omega q \nu + \lambda, \quad g(z) = \omega q^* \nu + \lambda$$

where ω, ν, and $\lambda \in \mathbb{H}$ are the quaternion constants.

Solution: Using product rule (12) and setting $\mu = 1$, we have

$$\frac{\partial f(q)}{\partial q} = \frac{\partial(\omega q \nu)}{\partial q} = \omega q \frac{\partial \nu}{\partial q} + \frac{\partial(\omega q)}{\partial q^\nu}\nu = \omega \frac{\partial q}{\partial q^\nu}\nu = \omega \Re(\nu).$$

In a similar manner, it follows that:

$$\begin{aligned}\frac{\partial f(q)}{\partial q^*} &= \frac{\partial(\omega q \nu)}{\partial q^*} \\ &= \omega q \frac{\partial \nu}{\partial q^*} + \frac{\partial(\omega q)}{\partial q^{\nu *}}\nu = \omega \frac{\partial q}{\partial q^{\nu *}}\nu = -\frac{1}{2}\omega \nu^* \\ \frac{\partial g(q)}{\partial q} &= \frac{\partial(\omega q^* \nu)}{\partial q} \\ &= \omega q^* \frac{\partial \nu}{\partial q} + \frac{\partial(\omega q^*)}{\partial q^\nu}\nu = \omega \frac{\partial q^*}{\partial q^\nu}\nu = -\frac{1}{2}\omega \nu^* \\ \frac{\partial g(q)}{\partial q^*} &= \frac{\partial(\omega q^* \nu)}{\partial q^*} \\ &= \omega q^* \frac{\partial \nu}{\partial q^*} + \frac{\partial(\omega q^*)}{\partial q^{\nu *}}\nu = \omega \frac{\partial q^*}{\partial q^{\nu *}}\nu = \omega \Re(\nu).\end{aligned}$$

Remark 5: Observe that for $\mu \in \{1, i, j, k\}$, the HR derivatives in (8) and (9) are a special case of the right GHR derivative in (11), the latter being more general, as the GHR derivatives incorporate the product and chain rules in (12)–(15). These rules are instrumental for compact and elegant quaternion optimization.

Remark 6: Due to the noncommutativity of quaternion products, in general, the left GHR derivatives are different from the right GHR derivatives. However, they will be equal if the function f is real valued [23], as is the case with the most frequently used mean square error (MSE)-based optimization. In the sequel, we therefore mainly focus on the left GHR derivatives, because their convenient properties (12)–(18) are consistent with physical intuition.

C. Quaternion Gradient

The existing quaternion gradient formulations are much more restricted than the standard real gradient, which has prevented systematic development of quaternion gradient-based optimization. Although the quaternion pseudogradient (real component-wise gradients combined) can be used [10], the calculation of the pseudogradient is cumbersome and tedious, making the derivation of optimization algorithms of quaternion variables very prone to errors. Another type of quaternion gradient with local analyticity has been proposed in [17], which allows for derivatives of polynomials and some elementary functions. However, the products and compositions of two local analytic functions are generally not local analytic. An elegant new approach is the HR calculus [18], which defines the HR gradient with respect to a quaternion vector variable and its involutions. However, the traditional product rule is not applicable to the HR gradient. A gradient based on quaternion involutions, called the Involution-gradient (I-gradient) [33], provides a generic generalization of the real and complex least mean square (LMS) algorithms; however, it does not admit the traditional product rule. The quaternion gradient proposed in this paper rectifies these issues, as it is based on the GHR calculus, which incorporates the novel product and chain rules (12)–(15). For rigor, we consider a general case of functions $f(\mathbf{q}) : \mathbb{H}^{N\times 1} \to \mathbb{H}$, where $\mathbf{q} = (q_1, q_2, \ldots, q_N)^T \in \mathbb{H}^{N\times 1}$.

Definition 7 (Quaternion Gradient): The two quaternion gradients of a function $f : \mathbb{H}^{N\times 1} \to \mathbb{H}$ are defined as

$$\nabla_{\mathbf{q}} f \triangleq \left(\frac{\partial f}{\partial \mathbf{q}}\right)^T = \left(\frac{\partial f}{\partial q_1}, \ldots, \frac{\partial f}{\partial q_N}\right)^T \in \mathbb{H}^{N\times 1}$$
$$\nabla_{\mathbf{q}^*} f \triangleq \left(\frac{\partial f}{\partial \mathbf{q}^*}\right)^T = \left(\frac{\partial f}{\partial q_1^*}, \ldots, \frac{\partial f}{\partial q_N^*}\right)^T \in \mathbb{H}^{N\times 1}.$$

Definition 8 (Quaternion Jacobian Matrix): The quaternion Jacobian matrices of $\mathbf{f} : \mathbb{H}^{N\times 1} \to \mathbb{H}^{M\times 1}$ are defined as

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}} = \begin{pmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_M}{\partial q_1} & \cdots & \frac{\partial f_M}{\partial q_N} \end{pmatrix}, \quad \frac{\partial \mathbf{f}}{\partial \mathbf{q}^*} = \begin{pmatrix} \frac{\partial f_1}{\partial q_1^*} & \cdots & \frac{\partial f_1}{\partial q_N^*} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_M}{\partial q_1^*} & \cdots & \frac{\partial f_M}{\partial q_N^*} \end{pmatrix}.$$

Note that the convention for two vectors $\mathbf{f} \in \mathbb{H}^{M\times 1}$ and $\mathbf{q} \in \mathbb{H}^{N\times 1}$, $\partial \mathbf{f}/\partial \mathbf{q}$ is a matrix for which the (m, n)th element is $(\partial f_m/\partial q_n)$, thus, the dimension of $\partial \mathbf{f}/\partial \mathbf{q}$ is $M \times N$.

D. Quaternion Hessian

Since a formal derivative of a function $f : \mathbb{H} \to \mathbb{H}$ is (wherever it exists) again a function from $\mathbb{H}$ to $\mathbb{H}$, it makes sense to take the GHR derivative of a GHR derivative, that is, a higher order GHR derivative. We next consider second-order quaternion derivatives of the form

$$\frac{\partial^2 f}{\partial q^\mu \partial q^\nu} = \frac{\partial}{\partial q^\mu}\left(\frac{\partial f}{\partial q^\nu}\right), \quad \frac{\partial^2 f}{\partial q^{\mu *} \partial q^{\nu *}} = \frac{\partial}{\partial q^{\mu *}}\left(\frac{\partial f}{\partial q^{\nu *}}\right)$$
$$\frac{\partial^2 f}{\partial q^\mu \partial q^{\nu *}} = \frac{\partial}{\partial q^\mu}\left(\frac{\partial f}{\partial q^{\nu *}}\right), \quad \frac{\partial^2 f}{\partial q^{\mu *} \partial q^{\nu}} = \frac{\partial}{\partial q^{\mu *}}\left(\frac{\partial f}{\partial q^{\nu}}\right).$$

Note that the second-order cross derivatives are in general not identical [23], that is $(\partial^2 f/\partial q^\mu \partial q^\nu) \neq (\partial^2 f/\partial q^\nu \partial q^\mu)$. However, the second-order GHR derivatives do commute [15], [23]

$$\frac{\partial^2 f}{\partial q^\mu \partial q^{\mu *}} = \frac{\partial^2 f}{\partial q^{\mu *} \partial q^\mu} = \frac{1}{16}\left(\frac{\partial^2 f}{\partial q_a^2} + \frac{\partial^2 f}{\partial q_b^2} + \frac{\partial^2 f}{\partial q_c^2} + \frac{\partial^2 f}{\partial q_d^2}\right). \tag{19}$$

For a real valued $f : \mathbb{H} \to \mathbb{R}$, the conjugate rule for the second-order GHR derivatives is given by [23]

$$\left(\frac{\partial^2 f}{\partial q^\mu \partial q^\nu}\right)^* = \frac{\partial^2 f}{\partial q^{\nu *} \partial q^{\mu *}}, \quad \left(\frac{\partial^2 f}{\partial q^{\mu *} \partial q^{\nu *}}\right)^* = \frac{\partial^2 f}{\partial q^{\nu} \partial q^{\mu}}$$
$$\left(\frac{\partial^2 f}{\partial q^\mu \partial q^{\nu *}}\right)^* = \frac{\partial^2 f}{\partial q^{\nu} \partial q^{\mu *}}, \quad \left(\frac{\partial^2 f}{\partial q^{\mu *} \partial q^{\nu}}\right)^* = \frac{\partial^2 f}{\partial q^{\nu *} \partial q^{\mu}}. \tag{20}$$

Definition 9 (Quaternion Hessian Matrix): Let $f : \mathbb{H}^{N\times 1} \to \mathbb{H}$, then the two quaternion Hessian matrices of the mapping f are defined as

$$\mathbf{H}_{\mathbf{qq}} \triangleq \frac{\partial}{\partial \mathbf{q}}\left(\frac{\partial f}{\partial \mathbf{q}}\right)^T = \begin{pmatrix} \frac{\partial^2 f}{\partial q_1 \partial q_1} & \cdots & \frac{\partial^2 f}{\partial q_N \partial q_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial q_1 \partial q_N} & \cdots & \frac{\partial^2 f}{\partial q_N \partial q_N} \end{pmatrix} \in \mathbb{H}^{N\times N}$$

$$\mathbf{H}_{\mathbf{qq}^*} \triangleq \frac{\partial}{\partial \mathbf{q}}\left(\frac{\partial f}{\partial \mathbf{q}^*}\right)^T = \begin{pmatrix} \frac{\partial^2 f}{\partial q_1 \partial q_1^*} & \cdots & \frac{\partial^2 f}{\partial q_N \partial q_1^*} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial q_1 \partial q_N^*} & \cdots & \frac{\partial^2 f}{\partial q_N \partial q_N^*} \end{pmatrix} \in \mathbb{H}^{N\times N}.$$

III. Optimization in the Quaternion Field

Consider the quaternion dynamic systems described by the nonlinear ordinary differential equation $\dot{q} = \Phi(q(t))$, where Φ is the quaternion nonlinearity and $q(t)$ is the state of quaternion variable. The objective function J is expressed in terms of the adjustable parameter vector $\mathbf{w}$ and the values of the state at discrete points $t - r, r = 0, 1, \ldots, N - 1$, that is, $J = J[\mathbf{q}, \mathbf{w}]$, where $\mathbf{q} = [q(t), q(t-1), \ldots, q(t-N+1)]^T$. The optimization problems for real functions of quaternion parameters frequently arise in engineering applications and can be formulated as

$$\min_{\mathbf{w}\in\mathbb{H}^N} J(\mathbf{w})$$

where J is a real smooth function in N quaternion variables $\mathbf{w}$. Solutions often require a first- or second-order approximation of the objective function to generate a new step or a descent direction. However, the quaternion analyticity conditions given in [15]–[17] assert that the real-valued function of quaternion variable is necessarily nonanalytic. A conventional approach for algorithm derivation is to cast the optimization problem into the real domain by separating the four real components of $\mathbf{w}$ and then taking the real derivatives. The consequence is that the computations may become cumbersome and tedious. The GHR calculus provides an alternative elegant formulation, which is based on simple rules and principles (12)–(18) and is a natural generalization of the CR derivatives [21], [24], [29].

For an intuitive link between the real and quaternion vectors, consider a quaternion vector $\mathbf{q} = \mathbf{q}_a + i\mathbf{q}_b + j\mathbf{q}_c + k\mathbf{q}_d \in \mathbb{H}^{N\times 1}$, expressed by its real coordinate vectors $\mathbf{q}_a, \mathbf{q}_b, \mathbf{q}_c$, and $\mathbf{q}_d \in \mathbb{R}^{N\times 1}$. Following an approach similar to that in [18], [23], and [24] for the case of scalar quaternions, we can now define an augmented quaternion vector $\mathbf{h} \in \mathbb{H}^{4N\times 1}$, as shown in (21). The relationship between the augmented quaternion vector $\mathbf{h} \in \mathbb{H}^{4N\times 1}$ and its dual-quadrivariate real vector $\mathbf{r} \in \mathbb{R}^{4N\times 1}$ is given by [35], [36]

$$\underbrace{\begin{pmatrix} \mathbf{q} \\ \mathbf{q}^i \\ \mathbf{q}^j \\ \mathbf{q}^k \end{pmatrix}}_{\mathbf{h}} = \underbrace{\begin{pmatrix} \mathbf{I}_N & i\mathbf{I}_N & j\mathbf{I}_N & k\mathbf{I}_N \\ \mathbf{I}_N & i\mathbf{I}_N & -j\mathbf{I}_N & -k\mathbf{I}_N \\ \mathbf{I}_N & -i\mathbf{I}_N & j\mathbf{I}_N & -k\mathbf{I}_N \\ \mathbf{I}_N & -i\mathbf{I}_N & -j\mathbf{I}_N & k\mathbf{I}_N \end{pmatrix}}_{\mathbf{J}} \underbrace{\begin{pmatrix} \mathbf{q}_a \\ \mathbf{q}_b \\ \mathbf{q}_c \\ \mathbf{q}_d \end{pmatrix}}_{\mathbf{r}} \quad (21)$$

where $\mathbf{I}_N$ is the $N \times N$ identity matrix, and $\mathbf{J}$ is the $4N \times 4N$ matrix in (21). Multiplying both sides of (21) by $(1/4)\mathbf{J}^H$ and noting that $(1/4)\mathbf{J}^H\mathbf{J} = \mathbf{I}_{4N}$, we have

$$\mathbf{r} = \frac{1}{4}\mathbf{J}^H\mathbf{h} \in \mathbb{R}^{4N\times 1}. \quad (22)$$

From (21), a real scalar function $f(\mathbf{q}) : \mathbb{H}^{N\times 1} \to \mathbb{R}$ can be viewed in three equivalent forms

$$f(\mathbf{q}) \Leftrightarrow f(\mathbf{q}_a, \mathbf{q}_b, \mathbf{q}_c, \mathbf{q}_d) \triangleq f(\mathbf{r}) \Leftrightarrow f(\mathbf{q}, \mathbf{q}^i, \mathbf{q}^j, \mathbf{q}^k) \triangleq f(\mathbf{h}). \quad (23)$$

Since (22) is a linear transformation and $\mathbf{r}$ is a real vector, it follows that:

$$\frac{\partial f}{\partial \mathbf{h}} = \frac{\partial f}{\partial \mathbf{r}}\frac{\partial \mathbf{r}}{\partial \mathbf{h}} = \frac{1}{4}\frac{\partial f}{\partial \mathbf{r}}\mathbf{J}^H \Leftrightarrow \frac{\partial f}{\partial \mathbf{r}} = \frac{\partial f}{\partial \mathbf{h}}\mathbf{J}, \quad f \in \mathbb{R} \quad (24)$$

where $(\partial f/\partial \mathbf{h}) \in \mathbb{H}^{1\times 4N}$ and $(\partial f/\partial \mathbf{r}) \in \mathbb{R}^{1\times 4N}$. Since f and $\mathbf{r}$ are real valued, we have

$$\begin{aligned} \nabla_{\mathbf{r}} f &\triangleq \left(\frac{\partial f}{\partial \mathbf{r}}\right)^T = \left(\frac{\partial f}{\partial \mathbf{r}}\right)^H \\ &= \left(\frac{\partial f}{\partial \mathbf{h}}\mathbf{J}\right)^H \qquad \text{(from (24))} \\ &= \mathbf{J}^H\left(\frac{\partial f}{\partial \mathbf{h}}\right)^H \\ &= \mathbf{J}^H\left(\frac{\partial f}{\partial \mathbf{h}^*}\right)^T \qquad \text{(from (18))} \\ &= \mathbf{J}^H\nabla_{\mathbf{h}^*} f. \end{aligned} \quad (25)$$

This shows that the real gradient $\nabla_{\mathbf{r}} f \in \mathbb{R}^{4N\times 1}$ and the augmented quaternion gradient $\nabla_{\mathbf{h}^*} f \in \mathbb{H}^{4N\times 1}$ are related by a simple invertible linear transformation $\mathbf{J}^H$.

Remark 10: Based on (21), (23), and (25), we can now state a necessary and sufficient condition for the existence of stationary points of a real-valued function f in the form of the following equivalent relations:

$$\frac{\partial f}{\partial \mathbf{q}} = \mathbf{0} \Leftrightarrow \frac{\partial f}{\partial \mathbf{q}^*} = \mathbf{0} \Leftrightarrow \frac{\partial f}{\partial \mathbf{r}} = \mathbf{0} \Leftrightarrow \frac{\partial f}{\partial \mathbf{h}} = \mathbf{0} \Leftrightarrow \frac{\partial f}{\partial \mathbf{h}^*} = \mathbf{0}.$$

A. Quaternion Gradient Descent Algorithm

Gradient descent (also known as steepest descent) is a first-order optimization algorithm, which finds a local minimum of a function by taking iterative steps proportional to the negative of the gradient of the function. From (23), a real scalar function $f(\mathbf{q}) : \mathbb{H}^{N\times 1} \to \mathbb{R}$, can also be viewed as $f(\mathbf{r}) : \mathbb{R}^{4N\times 1} \to \mathbb{R}$, for which the quadrivariate real gradient descent update rule is given by [37], [46]

$$\Delta\mathbf{r} = -\alpha\nabla_{\mathbf{r}} f, \quad \mathbf{r} \in \mathbb{R}^{4N\times 1} \quad (26)$$

where $\Delta\mathbf{r}$ denotes a small increment in $\mathbf{r}$ and $\alpha \in \mathbb{R}^+$ is the step size. Using (21), (25), and (26), we now obtain

$$\Delta\mathbf{h} = \mathbf{J}\Delta\mathbf{r} = -\alpha\mathbf{J}\nabla_{\mathbf{r}} f = -\alpha\mathbf{J}\mathbf{J}^H\nabla_{\mathbf{h}^*} f = -4\alpha\nabla_{\mathbf{h}^*} f. \quad (27)$$

From (21), we have $\mathbf{h} = (\mathbf{q}^T, \mathbf{q}^{iT}, \mathbf{q}^{jT}, \mathbf{q}^{kT})^T$, so that (27) can be rewritten as

$$\Delta\mathbf{h} = \begin{pmatrix} \Delta\mathbf{q} \\ \Delta\mathbf{q}^i \\ \Delta\mathbf{q}^j \\ \Delta\mathbf{q}^k \end{pmatrix} = -4\alpha\begin{pmatrix} \nabla_{\mathbf{q}^*} f \\ \nabla_{\mathbf{q}^{i*}} f \\ \nabla_{\mathbf{q}^{j*}} f \\ \nabla_{\mathbf{q}^{k*}} f \end{pmatrix}. \quad (28)$$

This gives the quaternion gradient descent update rule in the form

$$\Delta\mathbf{q} = -4\alpha\nabla_{\mathbf{q}^*} f = -4\alpha\left(\frac{\partial f}{\partial \mathbf{q}^*}\right)^T = -4\alpha\left(\frac{\partial f}{\partial \mathbf{q}}\right)^H, \quad f \in \mathbb{R} \quad (29)$$

where $\alpha \in \mathbb{R}^+$ is the step size, and the conjugation rule in (18) was used in the last equality above.

Remark 11: Note from (29) that the quaternion gradient of a real-valued scalar function f with respect to a quaternion vector $\mathbf{q}$ is equal to $\nabla_{\mathbf{q}^*} f = (\partial f/\partial \mathbf{q}^*)^T$, and not $\nabla_{\mathbf{q}} f$. This result is a generalization of the CR calculus given in [20], and makes a possible compact derivation of learning algorithms in $\mathbb{H}$.

B. Quaternion Taylor Series Expansion

Using (19) and (20), it follows that the matrix $\mathbf{H}_{\mathbf{q}\mathbf{q}^*}$ is Hermitian symmetric, so that $\mathbf{H}_{\mathbf{q}\mathbf{q}^*}^H = \mathbf{H}_{\mathbf{q}\mathbf{q}^*}$. Then, up to second order, the TSE of the real scalar function $f(\mathbf{q}) : \mathbb{H}^{N\times 1} \to \mathbb{R}$, when viewed as a real analytic function $f(\mathbf{r}) : \mathbb{R}^{4N\times 1} \to \mathbb{R}$ of the vector $\mathbf{r} \in \mathbb{R}^{4N\times 1}$ from (23), is given by [37], [46]

$$f(\mathbf{r} + \Delta\mathbf{r}) = f(\mathbf{r}) + \frac{\partial f}{\partial \mathbf{r}}\Delta\mathbf{r} + \frac{1}{2}\Delta\mathbf{r}^T\mathbf{H}_{\mathbf{r}\mathbf{r}}\Delta\mathbf{r} + \text{h.o.t} \quad (30)$$

where $\mathbf{H}_{\mathbf{r}\mathbf{r}} \triangleq \partial/\partial\mathbf{r}(\partial f/\partial \mathbf{r})^T \in \mathbb{R}^{4N\times 4N}$ is a real symmetric Hessian matrix, $\mathbf{H}_{\mathbf{r}\mathbf{r}}^T = \mathbf{H}_{\mathbf{r}\mathbf{r}}$, and h.o.t. denotes the higher order terms. From (21) and (24), the first-order term in the augmented quaternion space is calculated as

$$\frac{\partial f}{\partial \mathbf{r}}\Delta\mathbf{r} = \frac{1}{4}\frac{\partial f}{\partial \mathbf{h}}\mathbf{J}\mathbf{J}^H\Delta\mathbf{h} = \frac{\partial f}{\partial \mathbf{h}}\Delta\mathbf{h}. \quad (31)$$

Noting from (21) that $\mathbf{h} = (\mathbf{q}^T, \mathbf{q}^{iT}, \mathbf{q}^{jT}, \mathbf{q}^{kT})^T$, we can now expand the first-order term in (30) as follows:

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{r}}\Delta\mathbf{r} &= \frac{\partial f}{\partial \mathbf{h}}\Delta\mathbf{h} \quad \text{(from (31))} \\ &= \frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q} + \frac{\partial f}{\partial \mathbf{q}^i}\Delta\mathbf{q}^i + \frac{\partial f}{\partial \mathbf{q}^j}\Delta\mathbf{q}^j + \frac{\partial f}{\partial \mathbf{q}^k}\Delta\mathbf{q}^k \quad \text{(from (21))} \\ &= \frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q} + \left(\frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q}\right)^i + \left(\frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q}\right)^j + \left(\frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q}\right)^k \\ &= 4\Re\left\{\frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q}\right\} \quad \text{(from (22)).} \end{aligned} \quad (32)$$

Consider now the $4N \times 4N$ augmented quaternion Hessian matrix

$$\mathbf{H}_{\mathbf{hh}^*} \triangleq \frac{\partial}{\partial \mathbf{h}}\left(\frac{\partial f}{\partial \mathbf{h}^*}\right)^T = \begin{pmatrix} \mathbf{H}_{\mathbf{qq}^*} & \mathbf{H}_{\mathbf{q}^i\mathbf{q}^*} & \mathbf{H}_{\mathbf{q}^j\mathbf{q}^*} & \mathbf{H}_{\mathbf{q}^k\mathbf{q}^*} \\ \mathbf{H}_{\mathbf{qq}^{i*}} & \mathbf{H}_{\mathbf{q}^i\mathbf{q}^{i*}} & \mathbf{H}_{\mathbf{q}^j\mathbf{q}^{i*}} & \mathbf{H}_{\mathbf{q}^k\mathbf{q}^{i*}} \\ \mathbf{H}_{\mathbf{qq}^{j*}} & \mathbf{H}_{\mathbf{q}^i\mathbf{q}^{j*}} & \mathbf{H}_{\mathbf{q}^j\mathbf{q}^{j*}} & \mathbf{H}_{\mathbf{q}^k\mathbf{q}^{j*}} \\ \mathbf{H}_{\mathbf{qq}^{k*}} & \mathbf{H}_{\mathbf{q}^i\mathbf{q}^{k*}} & \mathbf{H}_{\mathbf{q}^j\mathbf{q}^{k*}} & \mathbf{H}_{\mathbf{q}^k\mathbf{q}^{k*}} \end{pmatrix}. \quad (33)$$

Its equivalence with $\mathbf{H}_{\mathbf{rr}} \in \mathbb{R}^{4N\times 4N}$ can be established as

$$\begin{aligned} \mathbf{H}_{\mathbf{rr}} &= \frac{\partial}{\partial \mathbf{r}}\left(\frac{\partial f}{\partial \mathbf{r}}\right)^T \\ &= \frac{\partial}{\partial \mathbf{r}}\left(\frac{\partial f}{\partial \mathbf{r}}\right)^H \quad \text{(since } f \text{ is real-valued)} \\ &= \frac{\partial}{\partial \mathbf{r}}\left(\frac{\partial f}{\partial \mathbf{h}}\mathbf{J}\right)^H \quad \text{(from (24))} \\ &= \frac{\partial}{\partial \mathbf{r}}\left\{\mathbf{J}^H\left(\frac{\partial f}{\partial \mathbf{h}}\right)^H\right\} \\ &= \frac{\partial}{\partial \mathbf{h}}\left\{\mathbf{J}^H\left(\frac{\partial f}{\partial \mathbf{h}}\right)^H\right\}\mathbf{J} \quad \text{(from (24))} \\ &= \mathbf{J}^H\frac{\partial}{\partial \mathbf{h}}\left(\frac{\partial f}{\partial \mathbf{h}^*}\right)^T\mathbf{J} \quad \text{(from (18))} \\ &= \mathbf{J}^H\mathbf{H}_{\mathbf{hh}^*}\mathbf{J}. \end{aligned} \quad (34)$$

Note that the Hermitian operator in (34) cannot be replaced with the transpose operator, because for quaternion matrices $(\mathbf{AB})^T \neq \mathbf{B}^T\mathbf{A}^T$. Recalling that the Hessian $\mathbf{H}_{\mathbf{rr}}$ is a real symmetric matrix, it is evident from (34) that $\mathbf{H}_{\mathbf{hh}^*}$ is Hermitian, that is, $\mathbf{H}_{\mathbf{hh}^*}^H = \mathbf{H}_{\mathbf{hh}^*}$. Subsequently, for the second-order term in (30), we have

$$\begin{aligned} \frac{1}{2}\Delta\mathbf{r}^T\mathbf{H}_{\mathbf{rr}}\Delta\mathbf{r} &= \frac{1}{2}\Delta\mathbf{r}^H\mathbf{H}_{\mathbf{rr}}\Delta\mathbf{r} \quad \text{(since } \mathbf{r} \text{ is real-valued)} \\ &= \frac{1}{2}\Delta\mathbf{r}^H\mathbf{J}^H\mathbf{H}_{\mathbf{hh}^*}\mathbf{J}\Delta\mathbf{r} \quad \text{(from (34))} \\ &= \frac{1}{2}\Delta\mathbf{h}^H\mathbf{H}_{\mathbf{hh}^*}\Delta\mathbf{h} \quad \text{(from (21)).} \end{aligned} \quad (35)$$

From (21) and (35), the second-order term in (30) can now be expanded as

$$\begin{aligned} \frac{1}{2}\Delta\mathbf{r}^T\mathbf{H}_{\mathbf{rr}}\Delta\mathbf{r} &= \frac{1}{2}\Delta\mathbf{h}^H\mathbf{H}_{\mathbf{hh}^*}\Delta\mathbf{h} \\ &= \frac{1}{2}\sum_{\mu,\nu\in\{1,i,j,k\}}(\Delta\mathbf{q}^\nu)^H\mathbf{H}_{\mathbf{q}^\mu\mathbf{q}^{\nu*}}\Delta\mathbf{q}^\mu \quad \text{(from (33))} \\ &= 2\sum_{\mu\in\{1,i,j,k\}}\Re(\Delta\mathbf{q}^H\mathbf{H}_{\mathbf{q}^\mu\mathbf{q}^*}\Delta\mathbf{q}^\mu) \quad \text{(from (22)).} \end{aligned} \quad (36)$$

Using (23), (30), (31), and (35), the TSE in $\mathbb{H}^{4N}$ (augmented TSE) up to the second-order term can be expressed as

$$f(\mathbf{h}+\Delta\mathbf{h}) = f(\mathbf{h}) + \frac{\partial f}{\partial \mathbf{h}}\Delta\mathbf{h} + \frac{1}{2}\Delta\mathbf{h}^H\mathbf{H}_{\mathbf{hh}^*}\Delta\mathbf{h} + \text{h.o.t.} \quad (37)$$

Finally, a combination of the expansions in (32) and (36) yields the TSE expressed directly in $\mathbb{H}^N$, given by

$$\begin{aligned} f(\mathbf{q}+\Delta\mathbf{q}) = f(\mathbf{q}) &+ 4\Re\left(\frac{\partial f}{\partial \mathbf{q}}\Delta\mathbf{q}\right) \\ &+ 2\sum_{\mu\in\{1,i,j,k\}}\Re\big(\Delta\mathbf{q}^H\mathbf{H}_{\mathbf{q}^\mu\mathbf{q}^*}\Delta\mathbf{q}^\mu\big) + \text{h.o.t.} \end{aligned} \quad (38)$$

Also, from (34) and noting that $(1/4)\mathbf{JJ}^H = \mathbf{I}_{4N}$, we have

$$\mathbf{H}_{\mathbf{hh}^*} - \lambda\mathbf{I}_{4N} = \frac{1}{16}\mathbf{J}\left(\mathbf{H}_{\mathbf{rr}} - 4\lambda\mathbf{I}_{4N}\right)\mathbf{J}^H. \quad (39)$$

Remark 12: Since the real scalar function f is nonanalytic, the TSE in (37) and (38) is always augmented, due to the presence of the terms $\Delta\mathbf{q}^\mu$, $\mu \in \{i, j, k\}$. This is in contrast to the complex TSE for analytic functions.

Remark 13: Equation (39) illustrates that the eigenvalues of the quadrivariate real Hessian $\mathbf{H}_{\mathbf{rr}}$ are quadruple of those for the augmented quaternion Hessian $\mathbf{H}_{\mathbf{hh}^*}$. An important consequence is that the augmented quaternion Hessian $\mathbf{H}_{\mathbf{hh}^*}$ and the quadrivariate real Hessian $\mathbf{H}_{\mathbf{rr}}$ have the same positive definiteness properties and condition number. This result is important in numerical applications using the Hessian such as quaternion Newton minimization.

C. Quaternion Newton Method

The Newton method is a second-order optimization method, which makes use of the Hessian matrix. This method often has better convergence properties than the gradient descent method, but it can be very expensive to calculate and store the Hessian matrix. For a real scalar cost function $f(\mathbf{q}) : \mathbb{H}^{N\times 1} \to \mathbb{R}$, which from (23) can be viewed as $f(\mathbf{r}) : \mathbb{R}^{4N\times 1} \to \mathbb{R}$, the Newton iteration step $\Delta\mathbf{r}$ for the minimization of the function $f(\mathbf{r})$ with respect to its real parameters $\mathbf{r} = (\mathbf{q}_a^T, \mathbf{q}_b^T, \mathbf{q}_c^T, \mathbf{q}_d^T)^T$ is described by [37], [46]

$$\mathbf{H}_{\mathbf{rr}}\Delta\mathbf{r} = -\nabla_{\mathbf{r}} f, \quad \mathbf{r} \in \mathbb{R}^{4N\times 1} \quad (40)$$

where $\nabla_{\mathbf{r}} f$ is the real gradient defined by (25). From (21), (25), (34), and (40), it then follows that:

$$\mathbf{J}^H\mathbf{H}_{\mathbf{hh}^*}\Delta\mathbf{h} = \mathbf{J}^H\mathbf{H}_{\mathbf{hh}^*}\mathbf{J}\Delta\mathbf{r} = \mathbf{H}_{\mathbf{rr}}\Delta\mathbf{r} = -\nabla_{\mathbf{r}} f = -\mathbf{J}^H\nabla_{\mathbf{h}^*} f. \quad (41)$$

Upon multiplying both sides of (41) by $(1/4)\mathbf{J}$ and noting that $(1/4)\mathbf{JJ}^H = \mathbf{I}_{4N}$, we obtain

$$\mathbf{H}_{\mathbf{hh}^*}\Delta\mathbf{h} = -\nabla_{\mathbf{h}^*} f \quad (42)$$

where $\mathbf{H}_{\mathbf{hh}^*}$ is the augmented quaternion Hessian defined by (33). Since $\mathbf{h} = (\mathbf{q}^T, \mathbf{q}^{iT}, \mathbf{q}^{jT}, \mathbf{q}^{kT})^T$, we can expand (42) as

$$\mathbf{H}_{\mathbf{hh}^*}\Delta\mathbf{h} = \mathbf{H}_{\mathbf{hh}^*}\begin{pmatrix} \Delta\mathbf{q} \\ \Delta\mathbf{q}^i \\ \Delta\mathbf{q}^j \\ \Delta\mathbf{q}^k \end{pmatrix} = -\begin{pmatrix} \nabla_{\mathbf{q}^*} f \\ \nabla_{\mathbf{q}^{i*}} f \\ \nabla_{\mathbf{q}^{j*}} f \\ \nabla_{\mathbf{q}^{k*}} f \end{pmatrix}. \quad (43)$$

If $\mathbf{H}_{\mathbf{hh}^*}$ [equivalently, $\mathbf{H}_{\mathbf{rr}}$ in (39)] is positive definite, then using the Banachiewicz inversion formula for the inverse of a nonsingular partitioned matrix [39], yields

$$\Delta \mathbf{h} = \begin{pmatrix} -\mathbf{H}_{\mathbf{qq}^*}^{-1} - \mathbf{L}\mathbf{T}^{-1}\mathbf{U} & \mathbf{L}\mathbf{T}^{-1} \\ \mathbf{T}^{-1}\mathbf{U} & -\mathbf{T}^{-1} \end{pmatrix} \begin{pmatrix} \nabla_{\mathbf{q}^*} f \\ \nabla_{\mathbf{q}^{i*}} f \\ \nabla_{\mathbf{q}^{j*}} f \\ \nabla_{\mathbf{q}^{k*}} f \end{pmatrix} \tag{44}$$

where $\mathbf{T} = \left(\mathbf{H}_{\mathbf{hh}^*}/\mathbf{H}_{\mathbf{qq}^*}\right)$ is the Schur complement [39] of $\mathbf{H}_{\mathbf{qq}^*}$ in $\mathbf{H}_{\mathbf{hh}^*}$, and

$$\mathbf{L} = \mathbf{H}_{\mathbf{qq}^*}^{-1} \begin{pmatrix} \mathbf{H}_{\mathbf{qq}^{i*}} \\ \mathbf{H}_{\mathbf{qq}^{j*}} \\ \mathbf{H}_{\mathbf{qq}^{k*}} \end{pmatrix}^H, \quad \mathbf{U} = \begin{pmatrix} \mathbf{H}_{\mathbf{qq}^{i*}} \\ \mathbf{H}_{\mathbf{qq}^{j*}} \\ \mathbf{H}_{\mathbf{qq}^{k*}} \end{pmatrix} \mathbf{H}_{\mathbf{qq}^*}^{-1}. \tag{45}$$

The invertibility of the Schur complement $\mathbf{T}$ follows from the positive definiteness of $\mathbf{H}_{\mathbf{hh}^*}$, so that the quaternion Newton update rule is given by:

$$\Delta \mathbf{q} = -\mathbf{H}_{\mathbf{qq}^*}^{-1} \nabla_{\mathbf{q}^*} f + \mathbf{L}\mathbf{T}^{-1} \begin{pmatrix} -\mathbf{H}_{\mathbf{qq}^{i*}} \mathbf{H}_{\mathbf{qq}^*}^{-1} \nabla_{\mathbf{q}^*} f + \nabla_{\mathbf{q}^{i*}} f \\ -\mathbf{H}_{\mathbf{qq}^{j*}} \mathbf{H}_{\mathbf{qq}^*}^{-1} \nabla_{\mathbf{q}^*} f + \nabla_{\mathbf{q}^{j*}} f \\ -\mathbf{H}_{\mathbf{qq}^{k*}} \mathbf{H}_{\mathbf{qq}^*}^{-1} \nabla_{\mathbf{q}^*} f + \nabla_{\mathbf{q}^{k*}} f \end{pmatrix}. \tag{46}$$

A substantial simplification can be introduced by avoiding the computation of the inverse of the Schur complement $\mathbf{T}$, so that the quaternion Newton method (QNM) in (46) may be approximated as

$$\Delta \mathbf{q} \approx -\mathbf{H}_{\mathbf{qq}^*}^{-1} \nabla_{\mathbf{q}^*} f. \tag{47}$$

Remark 14: Note that the QNM (46) and approximated QNM (47) directly operate in $\mathbb{H}^{N\times 1}$, thus removing the redundancy present in (44). For the interested reader, we leave an estimation problem of the upper bound of the approximation error between (46) and (47).

IV. Application Examples

The above introduced quaternion gradient and Hessian are essential for numerical solutions in quaternion-valued parameter estimation problems, and in quaternion learning systems.

A. Quaternion Least Mean Square

We can now derive the QLMS algorithm [10], [18], a workhorse of adaptive estimation, in an elegant and compact way using the GHR calculus. For convenience, the rather tedious standard QLMS derivation applied component wise is given in Appendix I. Within the QLMS, the cost function to be minimized is a real-valued function of quaternion-valued error

$$J(n) = |e(n)|^2 = e^*(n)e(n) \tag{48}$$

where the error $e(n) = d(n) - y(n)$, $d(n)$ is the desired signal, $y(n) = \mathbf{w}^T(n)\mathbf{x}(n)$ is the filter output, $\mathbf{w}(n) \in \mathbb{H}^{N\times 1}$ is the vector of filter coefficients, $\mathbf{x}(n) \in \mathbb{H}^{N\times 1}$ is the input vector, and N is the filter length.

From (29), the weight update of QLMS is given by

$$\mathbf{w}(n+1) - \mathbf{w}(n) = -\alpha \nabla_{\mathbf{w}^*} J(n) = -\alpha \left(\frac{\partial J(n)}{\partial \mathbf{w}(n)}\right)^H \tag{49}$$

where $\alpha > 0$ is the step size and the negative gradient $-\nabla_{\mathbf{w}^*} J(n)$ defines the direction of gradient descent in (29). To find $\nabla_{\mathbf{w}^*} J$, we use the product rule (12)

$$\frac{\partial J}{\partial \mathbf{w}} = e^* \frac{\partial e}{\partial \mathbf{w}} + \frac{\partial e^*}{\partial \mathbf{w}^e} e \tag{50}$$

where time index n is omitted for convenience. The above two partial derivatives now become

$$\begin{aligned} \frac{\partial e}{\partial \mathbf{w}} &= \frac{\partial (d - \mathbf{w}^T \mathbf{x})}{\partial \mathbf{w}} = -\frac{\partial (\mathbf{w}^T \mathbf{x})}{\partial \mathbf{w}} = -\Re(\mathbf{x}^T) \\ \frac{\partial e^*}{\partial \mathbf{w}^e} e &= \frac{\partial (d^* - \mathbf{x}^H \mathbf{w}^*)}{\partial \mathbf{w}^e} e = -\frac{\partial (\mathbf{x}^H \mathbf{w}^*)}{\partial \mathbf{w}^e} e = \frac{1}{2} \mathbf{x}^H e^* \end{aligned} \tag{51}$$

where the results $\partial(\mathbf{w}^T\mathbf{x})/\partial\mathbf{w}$ and $\partial(\mathbf{x}^H\mathbf{w}^*)/(\partial\mathbf{w}^e)e$ can be seen as a vector version of the GHR derivatives in Example 4. Substituting (51) into (50) now yields

$$\frac{\partial J}{\partial \mathbf{w}} = -e^* \Re(\mathbf{x}^T) + \frac{1}{2} \mathbf{x}^H e^* = -\frac{1}{2} \mathbf{x}^T e^*. \tag{52}$$

Finally, the update of the adaptive weight vector of QLMS becomes

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \alpha\, e(n) \mathbf{x}^*(n) \tag{53}$$

where the constant $1/2$ in (52) is absorbed into the step size α.

Remark 15: From (53) and (92), we can see that the QLMS derived using the GHR calculus is exactly the same as that using the pseudogradient. However, the derivation of component-wise gradient in Appendix I is too cumbersome and tedious. The expressions in (53) and (92) also provide a theoretical support for the gradient descent method in (29).

Remark 16: Note that if we start from $e(n) = d(n) - \mathbf{w}^H(n)\mathbf{x}(n)$, the final update rule of QLMS would become $\mathbf{w}(n+1) = \mathbf{w}(n) + \alpha\, \mathbf{x}(n) e^*(n)$. The QLMS algorithm in (53) is therefore a generic generalization of complex LMS [40] to the quaternion field.

Remark 17: The QLMS algorithm in (53) is different from the original QLMS [10] based on component-wise gradients, the HR-QLMS [18] based on the HR gradient, and the Involution-QLMS (I-QLMS) [33] based on the I-gradient. The difference from the original QLMS arises due to the rigorous use of the noncommutativity of quaternion product in (90) and (91). The difference from the HR-QLMS and I-QLMS is due to the rigorous use of the novel product rule in (50).

B. Quaternion Nonlinear Adaptive Filtering

We now derive the quaternion nonlinear gradient descent (QNGD) algorithm given in [27] according to the rules of GHR calculus. The same real-valued quadratic cost function as in LMS and CLMS is used, that is

$$J(n) = |e(n)|^2 \tag{54}$$

where $e(n) = d(n) - y(n)$ is the error between the desired signal $d(n)$ and the filter output $y(n) = \Phi(s(n))$, where $s(n) = \mathbf{w}^T(n)\mathbf{x}(n)$, Φ is the quaternion nonlinearity defined in Appendix II, $\mathbf{w}(n) \in \mathbb{H}^{N\times 1}$ forms filter coefficients, and $\mathbf{x}(n) \in \mathbb{H}^{N\times 1}$ defines the input vector.

From (29), the weight update is given by

$$\mathbf{w}(n+1) - \mathbf{w}(n) = -\alpha \nabla_{\mathbf{w}^*} J(n) = -\alpha \left(\frac{\partial J(n)}{\partial \mathbf{w}^*(n)} \right)^T \tag{55}$$

where $\alpha > 0$ is the step size. To find $\nabla_{\mathbf{w}^*} J$, we use the chain rule (15)

$$\frac{\partial J}{\partial \mathbf{w}^*} = \sum_{\mu \in \{1,i,j,k\}} \frac{\partial |e|^2}{\partial e^{\mu *}} \frac{\partial e^{\mu *}}{\partial \mathbf{w}^*} \tag{56}$$

where time index n is omitted for convenience. Using the product rule (13), the derivative of $|e|^2$ can be calculated as

$$\frac{\partial |e|^2}{\partial e^{\mu *}} = \frac{\partial (e^* e)}{\partial e^{\mu *}} = e^* \frac{\partial e}{\partial e^{\mu *}} + \frac{\partial e^*}{\partial e^{e \mu *}} e = \frac{1}{2} e^{\mu}. \tag{57}$$

Next, we need to calculate the following derivative:

$$\frac{\partial e^{\mu *}}{\partial \mathbf{w}^*} = \frac{\partial (d-y)^{\mu *}}{\partial \mathbf{w}^*} = -\frac{\partial y^{\mu *}}{\partial \mathbf{w}^*} = -\frac{\partial \Phi^{\mu *}(s)}{\partial \mathbf{w}^*}. \tag{58}$$

By the chain rule in (15), we have

$$\frac{\partial \Phi^{\mu *}(s)}{\partial \mathbf{w}^*} = \sum_{\nu \in \{1,i,j,k\}} \frac{\partial \Phi^{\mu *}(s)}{\partial s^{\nu *}} \frac{\partial s^{\nu *}}{\partial \mathbf{w}^*} \tag{59}$$

where the derivatives of $s^{\nu *}$ employ the term $\partial(\omega q^*)/\partial q^*$ in Example 4, to give

$$\frac{\partial s^*}{\partial \mathbf{w}^*} = \frac{\partial (\mathbf{x}^H \mathbf{w}^*)}{\partial \mathbf{w}^*} = \mathbf{x}^H. \tag{60}$$

Next, using the rotation rule (16) and the term $(\partial(\omega q^*)/\partial q^{\mu *})\mu$ in Example 4, we have

$$\begin{aligned}\frac{\partial s^{\nu *}}{\partial \mathbf{w}^*} &= \left(\frac{\partial s^*}{\partial \mathbf{w}^{\nu *}} \right)^{\nu} = -\nu \frac{\partial (\mathbf{x}^H \mathbf{w}^*)}{\partial \mathbf{w}^{\nu *}} \nu \\ &= -\nu \mathbf{x}^H \Re(\nu) = 0 \quad \forall \nu \in \{i,j,k\}.\end{aligned} \tag{61}$$

Substituting (60) and (61) into (59), then yields

$$\frac{\partial \Phi^{\mu *}(s)}{\partial \mathbf{w}^*} = \frac{\partial \Phi^{\mu *}(s)}{\partial s^*} \mathbf{x}^H \quad \forall \mu \in \{1,i,j,k\}. \tag{62}$$

By combining (57), (58), and (62) with (56), we finally obtain

$$\frac{\partial J}{\partial \mathbf{w}^*} = -\frac{1}{2} \sum_{\mu \in \{1,i,j,k\}} e^{\mu} \frac{\partial \Phi^{\mu *}(s)}{\partial s^*} \mathbf{x}^H. \tag{63}$$

This gives the update of the adaptive weight vector of QNGD algorithm in the form

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \alpha \sum_{\mu \in \{1,i,j,k\}} e^{\mu}(n) \frac{\partial \Phi^{\mu *}(s(n))}{\partial s^*(n)} \mathbf{x}^*(n) \tag{64}$$

where the constant $1/2$ in (63) is absorbed into the step size α.

Remark 18: For a linear function $\Phi(q) = q$, the QNGD algorithm will degenerate into the QLMS in Section IV-A. This shows that, as desired, the QLMS algorithm is a special case of the QNGD algorithm. In addition, using the GHR calculus for the QNGD algorithm, the nonlinear function Φ is not required to satisfy the odd-symmetry condition $\Phi^*(q) = \Phi(q^*)$, which is required in [27] and [28].

C. Quaternion Feedforward Neural Networks

For convenience, we shall consider a single hidden layer quaternion feedforward neural network. The forward equations for signal passing through the network are given by

$$\mathbf{y} = \mathbf{V}\mathbf{x} + \mathbf{a}, \quad \mathbf{h} = \Phi(\mathbf{y}), \quad \mathbf{g} = \mathbf{W}\mathbf{h} + \mathbf{b} \tag{65}$$

where Φ denotes the quaternion nonlinearity defined in Appendix II, $\mathbf{x}$ is the input signal, $\mathbf{h}, \mathbf{g}$ are, respectively, the output at hidden and output layers, $\mathbf{V}$ and $\mathbf{W}$ the weight matrices associated with hidden and output layer neurons, and $\mathbf{a}$ and $\mathbf{b}$ the biases at the hidden and output layer neurons. The network error produced at the output layer is defined by

$$\mathbf{e} = \mathbf{d} - \mathbf{g} \tag{66}$$

where $\mathbf{d}$ denotes the desired output vector. Then, the gradient descent algorithm minimizes a real-valued loss function

$$J = \|\mathbf{e}\|^2 = \mathbf{e}^H \mathbf{e}. \tag{67}$$

From (29), the GHR quaternion gradient of the error function is given by

$$\nabla_{\mathbf{q}^*} J = \left(\frac{\partial J}{\partial \mathbf{q}^*} \right)^T = \left(\frac{\partial J}{\partial \mathbf{q}} \right)^H. \tag{68}$$

Using the chain rule (14) and rotation rule (16), we have

$$\frac{\partial J}{\partial \mathbf{q}} = \sum_{\mu \in \{1,i,j,k\}} \frac{\partial \|\mathbf{e}\|^2}{\partial \mathbf{e}^{\mu}} \frac{\partial \mathbf{e}^{\mu}}{\partial \mathbf{q}} = -\sum_{\mu \in \{1,i,j,k\}} \frac{\partial \|\mathbf{e}\|^2}{\partial \mathbf{e}^{\mu}} (\mathbf{J}_{\mathbf{q}^{\mu}})^{\mu} \tag{69}$$

where $\mathbf{J}_{\mathbf{q}^{\mu}} \triangleq (\partial \mathbf{g}/\partial \mathbf{q}^{\mu})$ is the Jacobian matrix of $\mathbf{g}$, and the derivative of $\|\mathbf{e}\|^2$ is a vector version of the term $\partial |e|^2/\partial e^{\mu}$ in (57), given by

$$\frac{\partial \|\mathbf{e}\|^2}{\partial \mathbf{e}^{\mu}} = \frac{1}{2} (\mathbf{e}^{\mu})^H. \tag{70}$$

Substituting (69) and (70) into (68), we arrive at

$$\nabla_{\mathbf{q}^*} J = -\frac{1}{2} \sum_{\mu \in \{1,i,j,k\}} \left(\mathbf{J}_{\mathbf{q}^{\mu}}^H \mathbf{e} \right)^{\mu}. \tag{71}$$

Now, we shall derive the quaternion backpropagation (QBP) algorithm for the QVNN in an explicit form as

$$\mathbf{J}_{\mathbf{b}^{\mu}} = \delta_{\mu 1} \mathbf{I}, \quad \mathbf{J}_{\mathbf{y}^{\mu}} = \frac{\partial \mathbf{g}}{\partial \mathbf{y}^{\mu}} = \mathbf{W} \frac{\partial \mathbf{h}}{\partial \mathbf{y}^{\mu}} \tag{72}$$

where $\mathbf{I}$ is the identity matrix and $\delta_{\mu\nu}$ is the Kronecker delta

$$\delta_{\mu\nu} = \begin{cases} 1, & \mu = \nu \\ 0, & \mu \neq \nu. \end{cases} \tag{73}$$

We note from (65) that $\mathbf{J}_{\mathbf{a}^{\mu}} = \mathbf{J}_{\mathbf{y}^{\mu}}$. Thus, the update rules for the bias at hidden and output layers are

$$\Delta \mathbf{a} = \alpha \sum_{\mu \in \{1,i,j,k\}} \left(\mathbf{J}_{\mathbf{a}^{\mu}}^H \mathbf{e} \right)^{\mu}, \quad \Delta \mathbf{b} = \alpha \mathbf{e} \tag{74}$$

where $\alpha > 0$ is the learning rate. Using the chain rule (15), the update rules for hidden and output layer weight matrices become

$$\Delta \mathbf{V} = \Delta \mathbf{a}\, \mathbf{x}^H, \quad \Delta \mathbf{W} = \Delta \mathbf{b}\, \mathbf{h}^H. \tag{75}$$

Remark 19: A comparison of the weight update structure of the QBP (74), (75) with the complex backpropagation (CBP) [41], [42] shows that the QBP is not a simple extension of the CBP. The extra terms in the QBP weight update are needed to capture the additional statistical information that exists in the quaternion field.

D. Quaternion Recurrent Neural Networks

We next consider a three-layer quaternion recurrent neural network (QRNN) [43] with a concatenated input-feedback layer, a processing layer of computation nodes, and an output layer. Let $\mathbf{h}(n)$ and $\mathbf{g}(n)$ denote the output of processing and output layers at time index n, and $\mathbf{x}(n)$ denotes the external input vector. The following equations fully describe the QRNN:

$$\begin{aligned}\mathbf{y}(n) &= \mathbf{U}\,\mathbf{h}(n-1) + \mathbf{V}\mathbf{x}(n) + \mathbf{a}\\ \mathbf{h}(n) &= \Phi(\mathbf{y}(n)),\quad \mathbf{g}(n) = \mathbf{W}\mathbf{h}(n) + \mathbf{b}\end{aligned} \tag{76}$$

where Φ denotes the quaternion nonlinearity defined in Appendix II, $\mathbf{U}$ is the internal weight matrix, $\mathbf{V}$ and $\mathbf{W}$ are the input and output weight matrices, and $\mathbf{a}$ and $\mathbf{b}$ are the biases of the processing and output layers. The network error produced at the processing layer is defined by

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{g}(n) \tag{77}$$

where $\mathbf{d}(n)$ denotes the desired output. The objective of the network training is to minimize a real-valued loss function

$$J(n) = \|\mathbf{e}(n)\|^2 = \mathbf{e}^H(n)\mathbf{e}(n). \tag{78}$$

From (71), the quaternion gradient of the error function is given by

$$\nabla_{\mathbf{q}^*} J(n) = \left(\frac{\partial J(n)}{\partial \mathbf{q}}\right)^H = -\frac{1}{2}\sum_{\mu\in\{1,i,j,k\}} \left(\mathbf{J}^H_{\mathbf{q}^\mu}(n)\mathbf{e}(n)\right)^\mu \tag{79}$$

where $\mathbf{J}_{\mathbf{q}^\mu}(n) \triangleq (\partial \mathbf{g}(n)/\partial \mathbf{q}^\mu)$ is the Jacobian matrix of $\mathbf{g}(n)$. We now derive the quaternion real-time recurrent learning (QRTRL) algorithm for the QRNN in an explicit form as

$$\mathbf{J}_{\mathbf{b}^\mu}(n) = \delta_{\mu 1}\mathbf{I},\quad \mathbf{J}_{\mathbf{a}^\mu}(n) = \mathbf{W}\frac{\partial \mathbf{h}(n)}{\partial \mathbf{a}^\mu} \tag{80}$$

where $\delta_{\mu 1}$ is the Kronecker delta and $\mathbf{I}$ is the identity matrix. Using the chain rule (14) and rotation rule (16), we have

$$\begin{aligned}\frac{\partial \mathbf{h}(n)}{\partial \mathbf{a}} &= \sum_{\mu\in\{1,i,j,k\}} \frac{\partial \mathbf{h}(n)}{\partial \mathbf{y}^\mu(n)}\frac{\partial \mathbf{y}^\mu(n)}{\partial \mathbf{a}}\\ &= \sum_{\mu\in\{1,i,j,k\}} \frac{\partial \mathbf{h}(n)}{\partial \mathbf{y}^\mu(n)}\left(\frac{\partial \mathbf{y}(n)}{\partial \mathbf{a}^\mu}\right)^\mu\\ &= \sum_{\mu\in\{1,i,j,k\}} \mathbf{\Lambda}_{\mathbf{y}^\mu}(n)\left(\delta_{\mu 1}\mathbf{I} + \mathbf{U}\frac{\partial \mathbf{h}(n-1)}{\partial \mathbf{a}^\mu}\right)^\mu\end{aligned} \tag{81}$$

where the Jacobian $\mathbf{\Lambda}_{\mathbf{y}^\mu}(n) = (\partial \mathbf{h}(n)/\partial \mathbf{y}^\mu(n))$ is a diagonal matrix consisting of the derivatives of the activation function Φ. Note that, (81) is a recursive relation about $\partial \mathbf{h}(n)/\partial \mathbf{a}^\mu$ with the initial condition $\partial \mathbf{h}(n)/\partial \mathbf{a}^\mu = \mathbf{0}$ for $n \le 0$. Then, update rules for the bias at processing and output layers are

$$\Delta \mathbf{a}(n) = \alpha \sum_{\mu\in\{1,i,j,k\}} \left(\mathbf{J}^H_{\mathbf{a}^\mu}(n)\mathbf{e}(n)\right)^\mu,\quad \Delta \mathbf{b}(n) = \alpha\,\mathbf{e}(n) \tag{82}$$

where $\alpha > 0$ is the learning rate. Similar to (75), the update rules for processing layer weights can be written as

$$\begin{aligned}\Delta \mathbf{W}(n) &= \Delta \mathbf{b}(n)\mathbf{h}^H(n)\\ \Delta \mathbf{U}(n) &= \Delta \mathbf{a}(n)\mathbf{h}^H(n-1)\\ \Delta \mathbf{V}(n) &= \Delta \mathbf{a}(n)\mathbf{x}^H(n).\end{aligned} \tag{83}$$

Remark 20: The QRTRL in (81) is a generalization of CRTRL [44] to the quaternion case. The QRTRL does not impose restrictions on the type of activation function and all computations are directly performed in the quaternion field. The derivation can be straightforwardly extended to more complicated multilayered recurrent networks, such as the Jordan network.

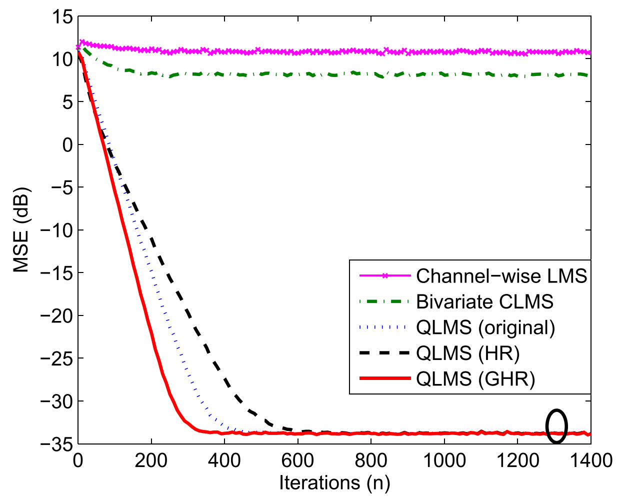
V. Simulations

To verify the suitability of the proposed quaternion gradient calculation for quaternion dynamic systems enabled by the novel GHR calculus, numerical simulations were conducted in the MATLAB programming environment.

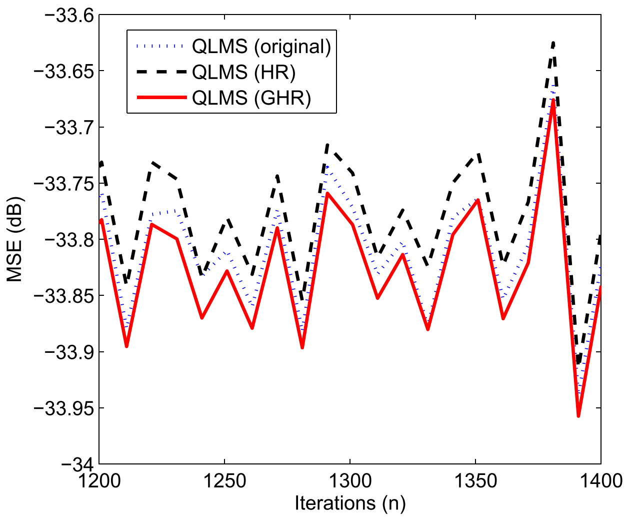
We first considered a performance comparison in the context of system identification of the QLMS derived by the proposed GHR calculus, the original QLMS, and the HR calculus-based QLMS [10], [18], as well as with two nonquaternion algorithms, the four-channel univariate LMS, and a pair of bivariate complex LMS. Note that both the channel-wise LMS and bivariate complex LMS adaptive filtering algorithms were suboptimal, but had reduced complexity as they dealt with quaternions as four/two independent real/complex quantities. The system coefficient vector $\mathbf{w}_o$ to be estimated was composed of uniformly distributed unit quaternions with length $N = 4$, given by

$$\mathbf{w}_o = \begin{bmatrix} 0.2806 + \imath 0.8315 + \jmath 0.4769 + \kappa 0.0502\\ 0.3914 + \imath 0.3184 + \jmath 0.8526 + \kappa 0.1359\\ 0.0469 + \imath 0.7521 + \jmath 0.4195 + \kappa 0.5061\\ 0.8895 + \imath 0.0627 + \jmath 0.1234 + \kappa 0.4355 \end{bmatrix}$$

while the system input vector $\mathbf{x}$ was generated using the quaternion-valued normal distribution. The desired signal $d(n)$ at time instant n was created by $d(n) = \mathbf{w}_o^T\mathbf{x}(n) + r(n)$, where $r(n)$ is the system noise, generated by quaternion-valued white Gaussian distribution and for which the variance was calculated so as to give the SNR at 40 dB. The step-size α was set to $\alpha = 0.005$ for all adaptive filtering algorithms considered. Fig. 1 shows the learning curves, measured in terms of the MSE averaged over 1000 trials. As expected, the nonquaternion adaptive filtering algorithms did not achieve acceptable performance due to the loss of mutual information among channels, and we also observed that among all QLMS versions in $\mathbb{H}$ the QLMS derived by the proposed GHR calculus, that is (53), exhibits both fastest convergence and smallest MSEs in the steady state, indicating that



(a)



(b)

Fig. 1. Evolution of the MSE $E(n)$ in decibel for the original QLMS, the HR calculus-based QLMS, and the proposed GHR calculus-based QLMS as well as their nonquaternion counterparts, four-channel univariate LMS and bivariate CLMS, averaged over 1000 trials. (a) MSE evolution process of all the considered algorithms. (b) Zoomed-in view in of the steady-state performance, elliptic region of Fig. 1(a).

the proposed GHR calculus enables more accurate gradient calculation as compared with the HR calculus.

We next performed a comprehensive comparison of the performances of a nonlinear finite-impulse response (FIR) filter trained by the proposed GHR calculus-based QNGD learning algorithm (64) and the original QNGD [27] in a one step-ahead signal prediction setting. The split-quaternion tanh function [27] was employed as the activation function, as defined in Appendix II and given in the following:

$$\Phi(q) = \tanh(q_a) + i\tanh(q_b) + j\tanh(q_c) + k\tanh(q_d) \quad (84)$$

where $q = q_a + iq_b + jq_c + kq_d \in \mathbb{H}$. The signal considered was the chaotic Lorenz signal, governed by the following coupled ordinary differential equations [45]:

$$\frac{\partial x}{\partial t} = \alpha(y-x), \quad \frac{\partial y}{\partial t} = x(\rho - z) - y, \quad \frac{\partial z}{\partial t} = xy - \beta z$$

where $\alpha = 10$, $\rho = 28$, and $\beta = 8/3$. The Lorenz attractor was modeled as a 3-D pure quaternion by mapping the three system states (x, y, and z) as $q = \imath x + \jmath y + \kappa z$. The coupled differential equations were recursively solved using the MATLAB function *ode45*, with the initial condition $x(0) = 5$, $y(0) = 5$, and $z(0) = 20$. To train the nonlinear FIR filter as a one step-ahead predictor, at time instant n, $\mathbf{q}(n) = [q(n), q(n-1), \ldots, q(n-N+1)]^T$ served as the input vector, and the difference between the output $y(k)$ and the desired signal $q(k+1)$ was used for the weight update in (64). The performance was measured using the prediction gain R_p defined as [46]

$$R_p = 10\log_{10}\frac{\hat{\sigma}_q^2}{\hat{\sigma}_e^2} \quad (85)$$

where $\hat{\sigma}_q^2$ and $\hat{\sigma}_e^2$ denote the estimated variances of the input and the prediction error. The length of the data sequence was set to 4000, and the last 500 samples were used to evaluate the prediction gain R_p. Fig. 2 shows the performance in terms of the prediction gains of both algorithms considered against different filter lengths N and step-sizes α. In all cases, the proposed GHR calculus provided a better prediction as compared with the original QNGD.

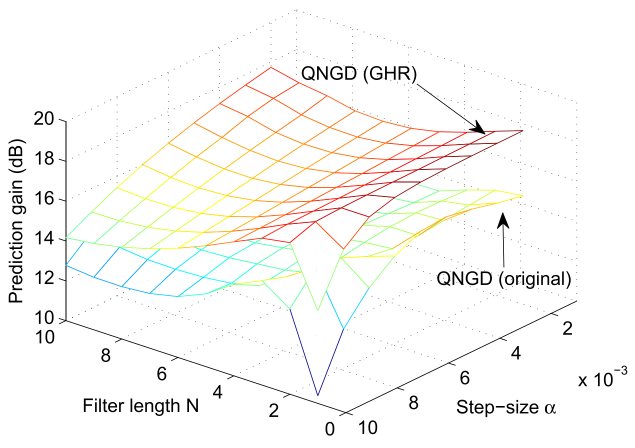


Fig. 2. Performance of QNGD versions on a one step-ahead prediction of the 3-D chaotic Lorenz attractor, against different step sizes and filter lengths.

The usefulness of the QRNN trained by the QRTRL algorithm, which is enabled by the proposed GHR calculus, was investigated based on a real-world 3-D nonstationary body motion tracking experiment. The 3-D motion data were recorded using the XSense MTx 3-DOF orientation tracker, placed on the left and right hands, left and right arms, and the waist of an athlete performing Tai Chi movements. The movement of the left arm was used as a pure quaternion input for this one step-ahead prediction experiment. The architecture of the QRNN consisted of 10 neurons with the tap input length $N = 5$; in this way, to predict the motion at time instant $n+1$, that is, $x(n+1)$, $\mathbf{x}(n) = [x(n), x(n-1), \ldots, x(n-N+1)]^T$ served as the input vector of the QRNN, as shown in (76). For comparison, the quaternion nonlinear FIR filtered with filter length $N = 5$ trained by the proposed GNGD algorithm was also used in this experiment, as it can be regarded as a simplified QRNN with a single input layer and an identity output matrix $\mathbf{W}$, as shown in (76). Fig. 3 shows that along all the three dimensions, the QRNN was able to track the

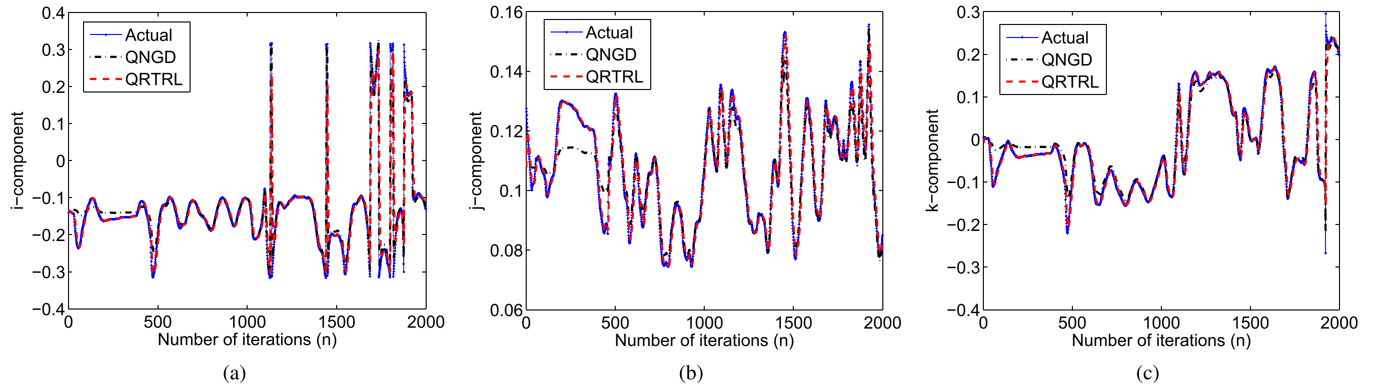


Fig. 3. Tracking capabilities of a nonlinear FIR filter trained by the QNGD algorithm and a QRNN trained by the QRTRL algorithm on a one step-ahead prediction of the 3-D left arm motion of an athlete performing Tai Chi movements. The two learning algorithms are enabled by the proposed GHR calculus. (a) $\imath$-component. (b) $\jmath$-component. (c) κ-component.

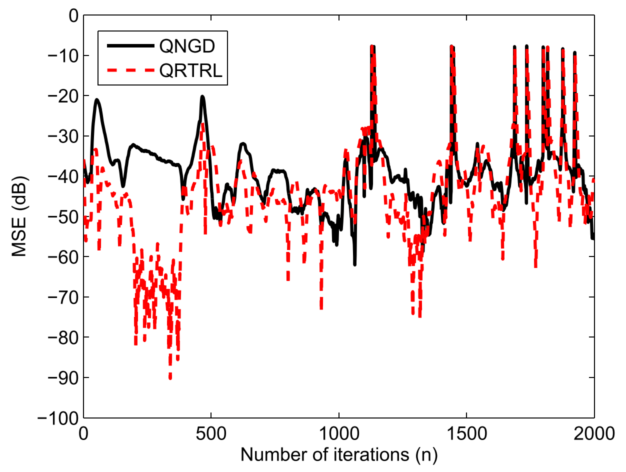


Fig. 4. MSE performance, evaluated over all the three dimensions, of the proposed QNGD and QRTRL algorithms for quaternion neural networks, for the real-world 3-D body motion tracking experiment.

left arm motion of the athlete more quickly and more accurately than the nonlinear FIR filter, which is further supported by the evolution of the MSE, as shown in Fig. 4.

VI. Conclusion

A new formulation for the quaternion gradient and Hessian of smooth real functions of quaternion variables has been proposed based on the novel GHR calculus. The QLMS, nonlinear adaptive filtering, backpropagation, and real-time recurrent learning algorithms have been derived in this way and shown to perform all computations directly in the quaternion field, without the need to increase the problem dimensionality. The GHR calculus thus resolves the long standing problems of quaternion analyticity, product, and chain rule, and greatly simplifies the derivation of first- and second-order iterative optimization procedures. The proposed framework has been shown to serve as a basis for generic extensions of real- and complex-valued optimization solutions. Illustrative simulations verify the faster convergence and better steady-state performance of the proposed algorithms over their counterparts enabled by the original HR calculus, as well as nonquaternion algorithms.

Appendix I
QLMS Derivation Using Component-Wise Pseudogradient

From Definitions 3 and 7, the weight update of QLMS can be written component wise as

$$\mathbf{w}(n+1)-\mathbf{w}(n) = -\alpha\nabla_{\mathbf{w}^*}J(n) = -\frac{1}{4}\alpha(\nabla_{\mathbf{w}_a}J(n) + \nabla_{\mathbf{w}_b}J(n)i + \nabla_{\mathbf{w}_c}J(n)j + \nabla_{\mathbf{w}_d}J(n)k) \tag{86}$$

where $\alpha > 0$ is the step size and the negative gradient $-\nabla_{\mathbf{w}^*}J(n)$ defines the direction of gradient descent in (29). Using the traditional product rule, the subgradients in (86) are calculated as

$$\begin{aligned}\nabla_{\mathbf{w}_a}J(n) &= e^*(n)(\nabla_{\mathbf{w}_a}e(n)) + (\nabla_{\mathbf{w}_a}e^*(n))e(n)\\ \nabla_{\mathbf{w}_b}J(n) &= e^*(n)(\nabla_{\mathbf{w}_b}e(n)) + (\nabla_{\mathbf{w}_b}e^*(n))e(n)\\ \nabla_{\mathbf{w}_a}J(n) &= e^*(n)(\nabla_{\mathbf{w}_c}e(n)) + (\nabla_{\mathbf{w}_c}e^*(n))e(n)\\ \nabla_{\mathbf{w}_d}J(n) &= e^*(n)(\nabla_{\mathbf{w}_d}e(n)) + (\nabla_{\mathbf{w}_d}e^*(n))e(n).\end{aligned} \tag{87}$$

The traditional product rule is valid here owing to the real valued nature of $\mathbf{w}_a, \mathbf{w}_b, \mathbf{w}_c$, and $\mathbf{w}_d$. We can now calculate the following subgradient:

$$\begin{aligned}&\nabla_{\mathbf{w}_a}e(n)\\ &= -\nabla_{\mathbf{w}_a}(\mathbf{w}^T(n)\mathbf{x}(n))\\ &= -\nabla_{\mathbf{w}_a}\big((\mathbf{w}_a^T(n)+\mathbf{w}_b^T(n)i+\mathbf{w}_c^T(n)j+\mathbf{w}_d^T(n)k)\mathbf{x}(n)\big)\\ &= -\nabla_{\mathbf{w}_a}\big(\mathbf{w}_a^T(n)\mathbf{x}(n)\big) = -\mathbf{x}(n).\end{aligned} \tag{88}$$

Similarly, the other terms in (87) can be obtained: $\nabla_{\mathbf{w}_b}e(n) = -i\mathbf{x}(n)$, $\nabla_{\mathbf{w}_c}e(n) = -j\mathbf{x}(n)$, and $\nabla_{\mathbf{w}_d}e(n) = -k\mathbf{x}(n)$. Following on (88), the subgradients of $e^*(n)$ in (87) can be expressed as:

$$\begin{aligned}&\nabla_{\mathbf{w}_a}e^*(n)\\ &= -\nabla_{\mathbf{w}_a}(\mathbf{x}^H(n)\mathbf{w}^*(n))\\ &= -\nabla_{\mathbf{w}_a}(\mathbf{x}^H(n)(\mathbf{w}_a(n)-\mathbf{w}_b(n)i-\mathbf{w}_c(n)j-\mathbf{w}_d(n)k))\\ &= -\nabla_{\mathbf{w}_a}(\mathbf{x}^H(n)\mathbf{w}_a(n)) = -\mathbf{x}^*(n).\end{aligned} \tag{89}$$

In a similar manner, we have $\nabla_{\mathbf{w}_b} e^*(n) = \mathbf{x}^*(n)i$, $\nabla_{\mathbf{w}_c} e^*(n) = \mathbf{x}^*(n)j$ and $\nabla_{\mathbf{w}_d} e^*(n) = \mathbf{x}^*(n)k$. Upon substituting (88) and (89) to (87), we arrive at

$$\begin{aligned}\nabla_{\mathbf{w}_a} J(n) &= -e^*(n)\mathbf{x}(n) - \mathbf{x}^*(n)e(n)\\ \nabla_{\mathbf{w}_b} J(n) &= -e^*(n)i\mathbf{x}(n) + \mathbf{x}^*(n)ie(n)\\ \nabla_{\mathbf{w}_c} J(n) &= -e^*(n)j\mathbf{x}(n) + \mathbf{x}^*(n)je(n)\\ \nabla_{\mathbf{w}_d} J(n) &= -e^*(n)k\mathbf{x}(n) + \mathbf{x}^*(n)ke(n). \end{aligned} \tag{90}$$

Substituting (90) to (86) yields

$$\begin{aligned}\nabla_{\mathbf{w}^*} J(n) &= -\frac{1}{4}e^*(n)(\mathbf{x}(n) + i\mathbf{x}(n)i + j\mathbf{x}(n)j + k\mathbf{x}(n)k)\\ &\quad -\frac{1}{4}\mathbf{x}^*(n)\left(e(n) - ie(n)i - je(n)j - ke(n)k\right)\\ &= \frac{1}{2}e^*(n)\mathbf{x}^*(n) - \mathbf{x}^*(n)\Re(e(n))\\ &= \left(\frac{1}{2}e^*(n) - \Re(e(n))\right)\mathbf{x}^*(n)\\ &= -\frac{1}{2}e(n)\mathbf{x}^*(n). \end{aligned} \tag{91}$$

Finally, we obtain the expression of the QLMS in the form

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \alpha\, e(n)\mathbf{x}^*(n) \tag{92}$$

where the constant $1/2$ in (91) is absorbed into the step size α.

APPENDIX II
DEFINITION OF QUATERNION NONLINEAR FUNCTIONS

One of the widely used quaternion nonlinear functions is the following split-quaternion function:

$$\Phi(q) = f_a(q_a) + if_b(q_b) + jf_c(q_c) + kf_d(q_d)$$

where $q = q_a + iq_b + jq_c + kq_d$ is the quaternion variable and $f_a, f_b, f_c, f_d : \mathbb{R} \to \mathbb{R}$ are the real-valued nonlinear differentiable functions, such as the hyperbolic tangent functions. Then, the GHR derivatives of the split-quaternion functions are given by

$$\begin{aligned}\frac{\partial\Phi(q)}{\partial q} &= \frac{1}{4}\left(\frac{\partial\Phi(q)}{\partial q_a} - \frac{\partial\Phi(q)}{\partial q_b}i - \frac{\partial\Phi(q)}{\partial q_c}j - \frac{\partial\Phi(q)}{\partial q_d}k\right)\\ &= \frac{1}{4}\left(f'_a(q_a) + f'_b(q_b) + f'_c(q_c) + f'_d(q_d)\right)\\ \frac{\partial\Phi(q)}{\partial q^*} &= \frac{1}{4}\left(\frac{\partial\Phi(q)}{\partial q_a} + \frac{\partial\Phi(q)}{\partial q_b}i + \frac{\partial\Phi(q)}{\partial q_c}j + \frac{\partial\Phi(q)}{\partial q_d}k\right)\\ &= \frac{1}{4}\left(f'_a(q_a) - f'_b(q_b) - f'_c(q_c) - f'_d(q_d)\right). \end{aligned}$$

A special case is when $f_a(x) = f_b(x) = f_c(x) = f_d(x) = \tanh(x)$. Then, the GHR derivatives of such a function become

$$\begin{aligned}\frac{\partial\Phi(q)}{\partial q} &= \frac{1}{4}\left(f'_a(q_a) + f'_b(q_b) + f'_c(q_c) + f'_d(q_d)\right)\\ &= \frac{1}{4}(4 - \tanh^2(q_a) - \tanh^2(q_b) - \tanh^2(q_c) - \tanh^2(q_d))\\ \frac{\partial\Phi(q)}{\partial q^*} &= \frac{1}{4}\left(f'_a(q_a) - f'_b(q_b) - f'_c(q_c) - f'_d(q_d)\right)\\ &= \frac{1}{4}(\tanh^2(q_b) + \tanh^2(q_c) + \tanh^2(q_d) - \tanh^2(q_a) - 2). \end{aligned}$$

This shows that the GHR derivatives of split-quaternion functions are real valued.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions on how to improve this paper.

REFERENCES

[1] N. Matsui, T. Isokawa, H. Kusamichi, F. Peperm, and H. Nishimura, "Quaternion neural network with geometrical operators," *J. Intell. Fuzzy Syst.*, vol. 15, nos. 3–4, pp. 149–164, Dec. 2004.

[2] S.-C. Pei and C.-M. Cheng, "Color image processing by using binary quaternion-moment-preserving thresholding technique," *IEEE Trans. Image Process.*, vol. 8, no. 5, pp. 614–628, May 1999.

[3] P. Arena, L. Fortuna, G. Muscato, and M. G. Xibilia, "Multilayer perceptrons to approximate quaternion valued functions," *Neural Netw.*, vol. 10, no. 2, pp. 335–342, Mar. 1997.

[4] L. Fortuna, G. Muscato, and M. G. Xibilia, "A comparison between HMLP and HRBF for attitude control," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 318–328, Mar. 2001.

[5] C. Jahanchahi and D. P. Mandic, "A class of quaternion Kalman filters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 533–544, Mar. 2014.

[6] F. A. Tobar and D. P. Mandic, "Quaternion reproducing kernel Hilbert spaces: Existence and uniqueness conditions," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5736–5749, Sep. 2014.

[7] Y. Xia, C. Jahanchahi, and D. P. Mandic, "Quaternion-valued echo state networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 4, pp. 663–673, Apr. 2015.

[8] Y. Xia, C. Jahanchahi, T. Nitta, and D. P. Mandic, "Performance bounds of quaternion estimators," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2015.2388782.

[9] F. Shang and A. Hirose, "Quaternion neural-network-based PolSAR land classification in Poincare-sphere-parameter space," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5693–5703, Sep. 2014.

[10] C. C. Took and D. P. Mandic, "The quaternion LMS algorithm for adaptive filtering of hypercomplex processes," *IEEE Trans. Signal Process.*, vol. 57, no. 4, pp. 1316–1327, Apr. 2009.

[11] C. C. Took and D. P. Mandic, "A quaternion widely linear adaptive filter," *IEEE Trans. Signal Process.*, vol. 58, no. 8, pp. 4427–4431, Aug. 2010.

[12] N. Le Bihan and J. Mars, "Singular value decomposition of quaternion matrices: A new tool for vector-sensor signal processing," *Signal Process.*, vol. 84, no. 7, pp. 1177–1199, Jul. 2004.

[13] S. Buchholz and N. Le Bihan, "Polarized signal classification by complex and quaternionic multi-layer perceptrons," *Int. J. Neural Syst.*, vol. 18, no. 2, pp. 75–85, Sep. 2008.

[14] A. J. Hanson, *Visualizing Quaternions*. San Francisco, CA, USA: Morgan Kaufmann, 2005.

[15] A. Sudbery, "Quaternionic analysis," *Math. Proc. Cambridge Philos. Soc.*, vol. 85, no. 2, pp. 199–225, 1979.

[16] C. A. Deavours, "The quaternion calculus," *Amer. Math. Monthly*, vol. 80, no. 9, pp. 995–1008, 1979.

[17] S. De Leo and P. P. Rotelli, "Quaternionic analyticity," *Appl. Math. Lett.*, vol. 16, no. 7, pp. 1077–1081, 2003.

[18] D. P. Mandic, C. Jahanchahi, and C. C. Took, "A quaternion gradient operator and its applications," *IEEE Signal Process. Lett.*, vol. 18, no. 1, pp. 47–50, Jan. 2011.

[19] W. Wirtinger, "Zur formalen theorie der funktionen von mehr komplexen veränderlichen," *Math. Ann.*, vol. 97, no. 1, pp. 357–375, 1927.

[20] D. H. Brandwood, "A complex gradient operator and its application in adaptive array theory," *IEE Proc. F, Commun., Radar Signal Process.*, vol. 130, no. 1, pp. 11–16, Feb. 1983.

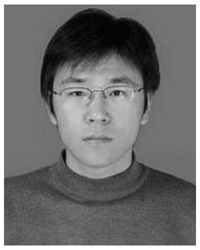
[21] K. Kreutz-Delgado. (2009). "The complex gradient operator and the CR-calculus." [Online]. Available: http://arxiv.org/abs/0906.4835

[22] T. A. Ell and S. J. Sangwine, "Quaternion involutions and anti-involutions," *Comput. Math. Appl.*, vol. 53, no. 1, pp. 137–143, 2007.

[23] D. Xu, C. Jahanchahi, C. C. Took, and D. P. Mandic. (2014). "Quaternion derivatives: The GHR calculus." [Online]. Available: http://arxiv.org/abs/1409.8168v1

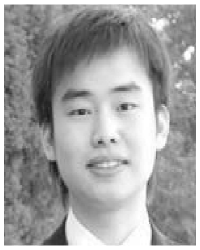
[24] A. van den Bos, "Complex gradient and Hessian," *IEE Proc.-Vis, Image Signal Process.*, vol. 141, no. 6, pp. 380–383, Dec. 2011.

[25] L. Sorber, M. Van Barel, and L. De Lathauwer, "Unconstrained optimization of real functions in complex variables," *SIAM J. Optim.*, vol. 22, no. 3, pp. 879–898, Jul. 2012.
[26] H. Li and T. Adali, "A class of complex ICA algorithms based on the kurtosis cost function," *IEEE Trans. Neural Netw.*, vol. 19, no. 3, pp. 408–420, Mar. 2008.
[27] B. C. Ujang, C. C. Took, and D. P. Mandic, "Split quaternion nonlinear adaptive filtering," *Neural Netw.*, vol. 23, no. 3, pp. 426–434, Apr. 2010.
[28] B. C. Ujang, C. C. Took, and D. P. Mandic, "Quaternion-valued nonlinear adaptive filtering," *IEEE Trans. Neural Netw.*, vol. 22, no. 8, pp. 1193–1206, Aug. 2011.
[29] D. P. Mandic and S. L. Goh, *Complex Valued Nonlinear Adaptive Filters: Noncircularity, Widely Linear and Neural Models*. New York, NY, USA: Wiley, 2009.
[30] J. P. Ward, *Quaternions and Cayley Numbers: Algebra and Applications*. Boston, MA, USA: Kluwer, 1997.
[31] F. Zhang, "Quaternions and matrices of quaternions," *Linear Algebra Appl.*, vol. 251, pp. 21–57, Jan. 1997.
[32] D. Xu and D. P. Mandic, "The theory of quaternon matrix derivatives," *IEEE Trans. Signal Process.*, vol. 63, no. 6, pp. 1543–1556, Mar. 2015.
[33] C. C. Took, C. Jahanchahi, and D. P. Mandic, "A unifying framework for the analysis of quaternion valued adaptive filters," in *Proc. Conf. Rec. 45th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2011, pp. 1771–1774.
[34] J. Navarro-Moreno, "ARMA prediction of widely linear systems by using the innovations algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3061–3068, Jul. 2008.
[35] C. C. Took and D. P. Mandic, "Augmented second-order statistics of quaternion random signals," *Signal Process.*, vol. 91, no. 2, pp. 214–224, Feb. 2011.
[36] J. Vía, D. Ramírez, and I. Santamaría, "Properness and widely linear processing of quaternion random vectors," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3502–3515, Jul. 2010.
[37] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer-Verlag, 1999.
[38] S. O. Haykin, *Adaptive Filter Theory*, 4th ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.
[39] F. Zhang, Ed., *The Schur Complement and Its Applications*. Dordrecht, The Netherlands: Springer-Verlag, 2005.
[40] B. Widrow, J. McCool, and M. Ball, "The complex LMS algorithm," *Proc. IEEE*, vol. 63, no. 4, pp. 719–720, Apr. 1975.
[41] H. Leung and S. Haykin, "The complex backpropagation algorithm," *IEEE Trans. Signal Process.*, vol. 39, no. 9, pp. 2101–2104, Sep. 1991.
[42] T. Nitta, "An extension of the back-propagation algorithm to complex numbers," *Neural Netw.*, vol. 10, no. 8, pp. 1391–1415, Nov. 1997.
[43] D. P. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*. New York, NY, USA: Wiley, 2001.
[44] S. L. Goh and D. P. Mandic, "An augmented CRTRL for complex-valued recurrent neural networks," *Neural Netw.*, vol. 20, no. 10, pp. 1061–1066, Dec. 2007.
[45] S. H. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering* (Studies in Nonlinearity). Boulder, CO, USA: Westview, 2001.
[46] S. Haykin and L. Li, "Nonlinear adaptive prediction of nonstationary signals," *IEEE Trans. Signal Process.*, vol. 43, no. 2, pp. 526–535, Feb. 1995.



Dongpo Xu received the B.S. degree in applied mathematics from Harbin Engineering University, Harbin, China, in 2004, and the Ph.D. degree in computational mathematics from the Dalian University of Technology, Dalian, China, in 2009.

He was a Lecturer with the College of Science, Harbin Engineering University, from 2009 to 2014, a Visiting Scholar with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., from 2013 to 2014. He is currently a Lecturer with the School of Mathematics and Statistics, Northeast Normal University, Changchun, China. His current research interests include neural networks, machine learning, and signal processing.



Yili Xia (M'11) received the B.Eng. degree in information engineering from Southeast University, Nanjing, China, in 2006, the M.Sc. (Hons.) degree in communications and signal processing from the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., in 2007, and the Ph.D. degree in adaptive signal processing from Imperial College London, in 2011.

He has been a Research Associate with Imperial College London since 2011. He is currently an Associate Professor with the School of Information and Engineering, Southeast University. His current research interests include linear and nonlinear adaptive filters, and complex valued and quaternion valued statistical analysis.

Danilo P. Mandic (M'99–SM'03–F'12) is currently a Professor of Signal Processing with Imperial College London, London, U.K., where he has been involved in nonlinear adaptive signal processing and nonlinear dynamics. He has been a Guest Professor with the Katholieke Universiteit Leuven, Leuven, Belgium, the Tokyo University of Agriculture and Technology, Tokyo, Japan, and Westminster University, London, and a Frontier Researcher with the RIKEN Brain Science Institute, Tokyo. His publications include two research monographs titled *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability* (Wiley, 2001) and *Complex Valued Nonlinear Adaptive Filters: Noncircularity, Widely Linear and Neural Models* (Wiley, 2009). He has edited a book titled *Signal Processing for Information Fusion* (Springer, 2008), and has over 200 publications on signal and image processing.

Prof. Mandic has been a member of the IEEE Technical Committee on Signal Processing Theory and Methods, and an Associate Editor of the *IEEE Signal Processing Magazine*, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, and the IEEE TRANSACTIONS ON NEURAL NETWORKS. He has produced award winning papers and products resulting from his collaboration with the industry.