Quaternion-Valued Adaptive Signal Processing and Its Applications to Adaptive Beamforming and Wind Profile Prediction

Mengdi Jiang

Supervisors:

Dr. Wei Liu

and

Dr. Yi Li

Thesis submitted in candidature

for graduating with degree of doctor of philosophy August 2016

Abstract

Quaternion-valued signal processing has received more and more attentions in the past ten years due to the increasing need to process three or four-dimensional signals, such as colour images, vector-sensor arrays, three-phase power systems, dual-polarisation based wireless communication systems, and wind profile prediction. One key operation involved in the derivation of all kinds of adaptive signal processing algorithms is the gradient operator. Although there are some derivations of this operator in literature with different level of details in the quaternion domain, it is still not fully clear how this operator can be derived in the most general case and how it can be applied to various signal processing problems. In this study, we will give a detailed derivation of the quaternion-valued gradient operator with associated properties and then apply it to different areas. In particular, it will be employed to derive the quaternion-valued LMS (QLMS) algorithm and its sparse versions for adaptive beamforming for vector sensor arrays, and another one is its application to wind profile prediction in combination with the classic computational fluid dynamics (CFD) approach.

For the adaptive beamforming problem for vector sensor arrays, we consider the crossed-dipole array and the problem of how to reduce the number of sensors involved in the adaptive beamforming process, so that reduced system complexity and energy consumption can be achieved, whereas an acceptable performance can still be maintained, which is particularly useful for large array systems. The quaternion-valued steering vector model for crossed-dipole arrays will be employed, and a reweighted zero attracting (RZA) QLMS algorithm is then proposed by introducing a RZA term to the cost function of the original QLMS algorithm. The RZA term aims to have a closer approximation to the l_0 norm so that the number of non-zero valued coefficients can be reduced more effectively in the adaptive beamforming process.

For wind profile prediction, it can be considered as a signal processing problem and we can solve it using traditional linear and non-linear prediction techniques, such as the proposed QLMS algorithm and its enhanced frequency-domain multi-channel version. On the other hand,

wind flow analysis is also a classical problem in the CFD field, which employs various simulation methods and models to calculate the speed of wind flow at different time. It is accurate but time-consuming with high computational cost. To tackle the problem, a combined approach based on synergies between the statistical signal processing approach and the CFD approach is proposed. There are different ways of combining the signal processing approach and the CFD approach to obtain a more effective and efficient method for wind profile prediction. In the combined method, the signal processing part employs the QLMS algorithm, while for the CFD part, large eddy simulation (LES) based on the Smagorinsky subgrid-scale (SGS) model will be employed so that more efficient wind profile prediction can be achieved.

Contents

List of Figures

List of Tables

List of Publications:

Journal papers:

1. M. D. Jiang, W. Liu and Y. Li, "Adaptive Beamforming for Vector-Sensor Arrays Based on Reweighted Zero-Attracting Quaternion-Valued LMS Algorithm ", *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, Issue 3, pp. 274-278, March 2016.

2. M. D. Jiang, Y. Li and W. Liu, "Properties of a General Quaternion-Valued Gradient Operator and Its Application to Signal Processing ", *Frontiers of Information Technology & Electronic Engineering*, vol. 17, issue 2, pp. 83-95, February 2016.

Conference papers:

1. M. D. Jiang, W. Liu and Y. Li, "A General Quaternion-valued Gradient Operator and Its Applications to Computational Fluid Dynamics and Adaptive Beamforming", *Proc. of the International Conference on Digital Signal Processing*, Hong Kong, August 2014.

2. M. D. Jiang, W. Liu and Y. Li, "A Zero-attracting Quaternion-valued Least Mean Square Algorithm for Sparse System Identification", *Proc. IEEE/IET International Sympoisum on Communication Systems, Networks and Digital Signal Processing*, Manchester, UK, July 2014.

3. M. D. Jiang, W. Liu, Y. Li and X. R. Zhang, “Frequency-domain Quaternion-valued Adaptive Filtering and Its Application to Wind Profile Prediction“, *Proc. of the IEEE TENCON Conference*, Xi’an,China, October 2013.

List of Abbreviations:

AQLMS	Augmented Quaternion-valued Least Mean Square
CFD	Computational Fluid Dynamics
DNS	Direct Numerical Simulation
DOA	Direction of Arrival
FFT	Fast Fourier Transform
HR	Hermitian Real-valued
LES	Large Eddy Simulation
LMS	Least Mean Square
MSE	Mean Square Error
QDFT	Quaternion-valued Discrete Fourier Transform
QLMS	Quaternion-valued Least Mean Square
RSB	Reference Signal Based
RZA-QLMS	Reweighted Zero-attracting QLMS
SGS	Sub-grid Scale Simulation
ULA	Uniform Linear Array
ZA-LMS	Zero-attracting LMS
ZA-QLMS	Zero-attracting QLMS

Acknowledgements:

I would like to take this opportunity to express my deepest gratitude to my supervisors Dr. Wei Liu and Dr. Yi Li for their persistant encouragement, guidance, support and for giving me the opportunity to pursue my doctoral studies under their supervision.

I would also like to thank National Grid (UK) for their financial support to the project, without which this work would not be possible. In particular, I am very grateful to Mr David Lenaghan for his patience, encouragement and valuable input to the project.

Finally, I would like to thank my family for their love and support during my study, which is beyond any word.

Chapter 1

Introduction

1.1 Introduction

Recently, hypercomplex concepts have been introduced to solve problems related to three or four-dimensional signals, such as vector-sensor array signal processing [1, 2, 3, 4], color image processing [5] and wind profile prediction [6, 7, 8]. In particular, it is widely used in wind prediction as the wind velocity has three orthogonal directions and the quaternionic model can meet this demand with three imaginary parts.

Therefore, some quaternion-valued signal processing methods and algorithms have been proposed already, such as the fast complexified quaternion Fourier transform [9] and quaternion-valued singular value decomposition [4]. Furthermore, in many of the cases, the traditional complex-valued adaptive filtering operation needs to be extended to the quaternion domain to derive the corresponding adaptive algorithms, such as the quaternion-valued Least Mean Square (QLMS) algorithm in [10]. One key operation involved in derivation of quaternion-valued adaptive algorithms is the gradient operator. Although there are some derivations of this operator in literature with different level of details, it is still not fully clear how this operator can be derived in the most general case and how it can be applied to various signal processing problems. Therefore, a general derivation of the quaternion-valued gradient operator applicable to different cases is needed, based on which the derived quaternion-valued algorithms can be implemented in different applications. In this thesis, we have considered two such applications: one is wind profile prediction combined with the classic computational fluid dynamics (CFD) approach, and the other one is the low-complexity adaptive beamforming problem for vector-sensor arrays.

1.2 Original Contributions

1.2.1 A general HR gradient operator and its applications

Notwithstanding the advantages of the quaternionic algorithms, extra care has to be taken in their developments, in particular when the derivatives of quaternion-valued functions are involved, due to the fact that quaternion algebra is non-commutative.

Regarding this, we first provide a detailed derivation of the relationship between the gradient and the increment of a quaternion function, highlighting the difference between the left and right gradients due to the non-commutativity of quaternion algebra. Secondly, we document several properties of the operators that have not been reported before, in particular several different versions of product rules and chain rules. Thirdly, we derive a general formula for the restricted HR derivatives of a wide class of regular quaternion-valued nonlinear functions, among which are the exponential, logarithmic, and the hyperbolic tangent functions. After that, we prove that the restricted HR gradients are consistent with the usual definition for the gradient of a real function in a real variable. Finally, we have derived some quaternion-valued adaptive algorithms based on the proposed general gradient operator.

1.2.2 Application to adaptive beamforming

In this work, we consider the crossed-dipole array and study the problem of how to reduce the number of sensors involved in the adaptive beamforming process so that reduced system complexity and energy consumption can be achieved, while an acceptable performance can still be maintained, which is especially useful for large array systems. In particular, we will use the quaternion-valued steering vector model for crossed-dipole arrays [2, 4, 11, 12, 13, 14, 15, 16, 17, 18], and propose a novel quaternion-valued adaptive algorithm for reference signal based adaptive beamforming.

In particular, based on recent advances in quaternion-valued signal processing, we derive a reweighted zero attracting (RZA) QLMS algorithm by introducing a RZA term to the cost function of the QLMS algorithm. Similar to the idea of the RZA least mean square (RZA-LMS) algorithm proposed in [19], the RZA term aims to have a closer approximation to the l_0 norm so that the number of non-zero valued coefficients can be reduced more effectively in the adaptive beamforming process.

1.2.3 Application to wind profile prediction

Wind profile prediction can be considered as a signal processing problem and we can solve it using traditional linear and non-linear prediction techniques, such as the proposed QLMS algorithm and its enhanced frequency-domain multi-channel version. On the other hand, wind flow analysis is also a classical problem in the CFD field, which employs various simulation methods and models to calculate the speed of wind flow at different time. It is accurate but time-consuming with high computational cost. To tackle the problem, a combined approach based on synergies between the statistical signal processing approach and the CFD approach is proposed.

There are different ways of combining the signal processing approach and the CFD approach to obtain a more effective and efficient method for wind profile prediction. In our current study, we mainly focus on the issue of efficiency, i.e. we aim to develop a method which can achieve a similar level of accuracy as the CFD approach but with a lower complexity. Certainly, it is possible to increase the complexity of the new method a little (but still lower than the original CFD approach) and achieve a more accurate result. In this case the new method could be more efficient and at the same time more effective as well. In the combined method, the signal processing part employs the QLMS algorithm, while for the CFD part, large eddy simulation (LES) based on the Smagorinsky subgrid-scale (SGS) model will be employed.

1.3 Outline

This thesis is organised as follows.

The algebra of quaternions as well as a brief review of previous research for both quaternion-valued gradient operators and the derived adaptive algorithms will be provided in Chapter 2. The general quaternion-valued gradient operator and some related chain rules and product rules will be presented in Chapter 3, where application of the gradient operator to the derivation of the QLMS adaptive algorithm and a nonlinear adaptive algorithm based on the hyperbolic tangent function is also discussed. At the final part of this chapter, a quaternion-valued adaptive algorithm is proposed for more efficient identification of unknown sparse systems, which is derived by introducing an l_1 penalty term in the original cost function and the resultant ZA-QLMS algorithm can achieve a faster convergence rate by incorporating the sparsity information of the system into the update process.

In Chapter 4, a review of the area of adaptive beamforming based on linear arrays will be provided first, and then the RZA-QLMS adaptive algorithm is proposed for low-complexity/cost adaptive beamforming based on vector sensor arrays consisting of crossed dipoles. It can reduce the number of sensors involved in the beamforming process so that reduced system complexity and energy consumption can be achieved while an acceptable performance can still be maintained, which is especially useful for large array systems.

Application of quaternion-based signal processing to wind profile prediction will be presented in Chapter 5, where correlation analysis will be performed to show that the problem can be dealt with by classic signal processing methods and then a multi-channel frequency-domain QLMS algorithm will be employed for wind profile prediction based on both recorded Google data and data generated by the Computational Fluid Dynamics (CFD) approach.

In Chapter 6, basic concepts related to CFD will be introduced and a combination of the CFD method and the quaternion-valued signal processing method will be described for efficient wind profile prediction. Conclusions are drawn in Chapter 7 with possible topics for future research.

Chapter 2

Quaternion-valued Adaptive Signal Processing

Adaptive filtering has been studied extensively in the past due to its applications in a wide range of areas, such as noise reduction, echo cancelation, beamforming, and speech coding, etc. [20, 21, 22, 23]. Initially, it was proposed for real-valued applications, and then extended to the complex domain to deal with bivariate signals particularly in digital communications [24]. Then, quaternion calculus was introduced in signal processing with application areas involving three or four-dimensional signals, such as color image processing [5, 25, 26, 27, 28], vector-sensor array systems [2, 4, 12, 13, 15, 16, 29], three-phase power systems [30], quaternion-valued wireless communications [31] and wind profile prediction [10].

In this chapter, we will first give an introduction to quaternion algebra and then review the complex-valued gradient operator and the derivation of complex-valued LMS algorithm. A quaternion-valued gradient operator based on real-valued cost function and the associated QLMS algorithm is presented in the following and the quaternion-valued discrete Fourier transform is introduced at the end.

2.1 Introduction to Quaternion Algebra

Quaternion is a non-commutative extension of complex numbers introduced by Hamilton [32]. A quaternion q is composed of four parts, i.e.,

$$q = q_a + q_b i + q_c j + q_d k \ , \tag{2.1}$$

where q_a is the real part, also denoted as $R(q)$. The other three terms constitute the imaginary part $I(q)$. i, j and k are the three imaginary units, satisfying the following rules:

$$ij = k, jk = i, ki = j, \tag{2.2}$$

$$i^2 = j^2 = k^2 = -1, \tag{2.3}$$

$$ij = -ji, ki = -ik, kj = -jk \tag{2.4}$$

As a result, in general the product of two quaternions p and q depends on the order, i.e., $qp \neq pq$, unless at least one of the factors is real-valued.

Let $v = |I(q)|$ and $\hat{\mathbf{v}} = I(q)/v$, the quaternion q can also be written as

$$q = q_a + v\hat{\mathbf{v}}, \tag{2.5}$$

where $\hat{\mathbf{v}}$ is a pure unit quaternion, which has the convenient property $\hat{\mathbf{v}}^2 := \hat{\mathbf{v}}\hat{\mathbf{v}} = -1$. The quaternionic conjugate of q is $q^* = q_a - q_b i - q_c j - q_d k$, or

$q^* = q_a - v\hat{\mathbf{v}}$. It is easy to show that

$$qq^* = q^*q = |q|^2, \tag{2.6}$$

$$q^{-1} = q^*/|q|^2. \tag{2.7}$$

To express the four real components of q, it is convenient to use its involutions $q^\nu := -\nu q \nu$ where $\nu \in \{i, j, k\}$ [33]. Explicitly, we have

$$q^i = -iqi = q_a + q_b i - q_c j - q_d k, \tag{2.8}$$

$$q^j = -jqj = q_a - q_b i + q_c j - q_d k, \tag{2.9}$$

$$q^k = -kqk = q_a - q_b i - q_c j + q_d k. \tag{2.10}$$

$$q_a = \frac{1}{4}(q + q^i + q^j + q^k), \tag{2.11}$$

$$q_b = \frac{1}{4i}(q + q^i - q^j - q^k), \tag{2.12}$$

$$q_c = \frac{1}{4j}(q - q^i + q^j - q^k), \tag{2.13}$$

$$q_d = \frac{1}{4k}(q - q^i - q^j + q^k). \tag{2.14}$$

Two useful relations are

$$q^* = \frac{1}{2}(q^i + q^j + q^k - q),$$

$$q + q^i + q^j + q^k = 4R(q). \tag{2.15}$$

2.2 Differentiation with Respect to a Vector in Complex Domain

Differentiation of a function with respect to a general complex-valued vector is a common problem in many areas of signal processing including array signal processing. Now a brief review about this topic is given as follows.

Assume that the function $f(z)$ is a function of the complex variable z, which can be expressed as

$$\begin{aligned} z &= x + jy \\ f(z) &= m + jn. \end{aligned} \tag{2.16}$$

Thus, the complex-valued differentiation df/dz at the point $z = z_0$ can be defined as

$$\begin{aligned} \frac{\mathrm{d}f}{\mathrm{d}z} &= \lim_{\Delta z \to 0} \frac{f(z_0 + \Delta z) - f(z_0)}{\Delta_z} \\ &= \lim_{\Delta x, \Delta y \to 0} \frac{\Delta m + j\Delta n}{\Delta x + j\Delta y} \end{aligned} \tag{2.17}$$

To find out the gradient property, we can set $\Delta x = 0$ and let $\Delta y \to 0$, then the following result can be obtained [22]

$$\frac{\mathrm{d}f}{\mathrm{d}z} = -j\frac{\partial f}{\partial y}, \tag{2.18}$$

or let $\Delta x \to 0$ with $\Delta y = 0$, we have

$$\frac{\mathrm{d}f}{\mathrm{d}z} = \frac{\partial f}{\partial x}, \tag{2.19}$$

which shows the definition for a general function $f(z)$ when both its real part m and its imaginary part n are differentiable [34]:

$$\frac{\mathrm{d}f}{\mathrm{d}z} = \frac{1}{2}(\frac{\partial f}{\partial x} - j\frac{\partial f}{\partial y}). \tag{2.20}$$

As to the derivative of $f(z)$ with respect to z^*, we can define it in a similar way shown below [34]:

$$\frac{\mathrm{d}f}{\mathrm{d}z^*} = \frac{1}{2}(\frac{\partial f}{\partial x} + j\frac{\partial f}{\partial y}). \tag{2.21}$$

An essential result with this definition is that the complex variable z is independent of its conjugate z^*, i.e.

$$\frac{\mathrm{d}z}{\mathrm{d}z^*} = \frac{\mathrm{d}z^*}{\mathrm{d}z} = 0 \tag{2.22}$$

When the complex variable z is replaced by a complex-valued vector

$$\mathbf{w} = [w_1\, w_2\, \cdots\, w_M]^T, \tag{2.23}$$

where $w_m = a_m + b_m j$, $m = 1, ..., M$, the differentiation of $f(\mathbf{w})$ with respect to $\mathbf{w}$ can be derived as follows [21]

$$\frac{\partial f}{\partial \mathbf{w}} = \frac{1}{2}\begin{bmatrix} \frac{\partial f}{\partial a_1} - j\frac{\partial f}{\partial b_1} \\ \frac{\partial f}{\partial a_2} - j\frac{\partial f}{\partial b_2} \\ \vdots \\ \frac{\partial f}{\partial a_M} - j\frac{\partial f}{\partial b_M} \end{bmatrix} \tag{2.24}$$

Similarly, we define $\frac{\partial f}{\partial \mathbf{w}^*}$ as [21]

$$\frac{\partial f}{\partial \mathbf{w}^*} = \frac{1}{2}\begin{bmatrix} \frac{\partial f}{\partial a_1} + j\frac{\partial f}{\partial b_1} \\ \frac{\partial f}{\partial a_2} + j\frac{\partial f}{\partial b_2} \\ \vdots \\ \frac{\partial f}{\partial a_M} + j\frac{\partial f}{\partial b_M} \end{bmatrix} \tag{2.25}$$

When $M = 1$, Eq. (2.24) and Eq. (2.25) are reduced to Eq. (2.20) and Eq. (2.21). Similarly, $\mathbf{w}$ and $\mathbf{w}^*$ are also independent of each other [22].

2.3 Complex-valued Least Mean Square Algorithm

In this section, based on the complex-valued gradient operator defined earlier, we review a widely used adaptive algorithm called the least mean square (LMS) algorithm introduced by Widrow and Hoff [35]. A significant feature of the LMS algorithm is its simplicity and due to its popularity and effectiveness, it has become the standard against which other linear adaptive filtering algorithms are evaluated [36, 37]. There are two fundamental steps in the algorithm: filtering and update, which are formed into a feedback loop [38].

The standard structure of the adaptive filter is shown in Fig. 2.1 [21, 39]. The output $y[n]$ is expressed as

$$y[n] = \mathbf{w}^H[n]\mathbf{x}[n], \tag{2.26}$$

and the error signal $e[n]$ is the difference between reference signal $d[n]$ and filter output, which is given by

$$\begin{aligned} e[n] &= d[n] - y[n] \\ &= d[n] - \mathbf{w}^H[n]\mathbf{x}[n], \end{aligned} \tag{2.27}$$

where $\mathbf{w}[n] = [w_1[n], w_2[n], \cdots, w_L[n]]^T$ is the adaptive weight vector with a length of L, $\mathbf{x}[n] = [x[n], x[n-1], \cdots, x[n-L+1]]^T$ is the input data vector with a size of $L \times 1$, and $\{\cdot\}^T$ denotes the transpose operation. The error signal $e[n]$ is employed for adjusting the weight vector $\mathbf{w}[n]$ according to some criterion, which is normally about minimising the error in a mean square or weighted sum of squares sense [21].

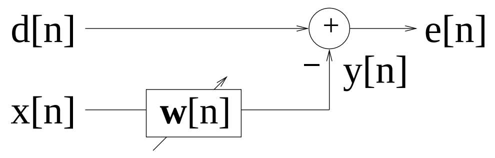

Fig. 2.1: A standard adaptive filter structure.

The cost function $J_0[n]$ of the LMS algorithm is constructed by the mean square error (MSE) and can be formulated as

$$\begin{aligned} J_0[n] &= E\{|e[n]|^2\} = E\{(d[n] - \mathbf{w}^H[n]\mathbf{x}[n])^2\} \\ &= \sigma_{dd}^2 - \mathbf{w}^H[n]\mathbf{p} - \mathbf{p}^H\mathbf{w}[n] + \mathbf{w}^H[n]\mathbf{R}_{xx}\mathbf{w}[n] \end{aligned} \tag{2.28}$$

where $\sigma_{dd}^2 = E\{|d[n]|^2\}$, $\mathbf{p} = E\{x[n]d^*[n]\}$ and $\mathbf{R}_{xx} = E\{x[n]x^H[n]\}$. The minimum value of the cost function can be found through updating the weight vector $\mathbf{w}[n]$ successively from an initial vector in the direction of the negative gradient of the MSE [40, 41], which is given by

$$\mathbf{w}[n+1] = \mathbf{w}[n] - \mu_0 \nabla J_0[n]. \tag{2.29}$$

The factor μ_0 is called the step size parameter, which is a positive real-valued constant weighting the amount of changes applied to each step. The gradient $\nabla J_0[n]$ with respect to $\mathbf{w}^*$ is expressed as

$$\nabla J_0[n] = \frac{\partial J_0[n]}{\partial \mathbf{w}^*}. \tag{2.30}$$

To calculate the gradient, we have to find out the exact second order statistics of the received signals, such as the cross-correlation vector $\mathbf{p}$ and the auto-correlation matrix $\mathbf{R}_{xx}$. However, it is impossible in practice since we have to use the known data to estimate the second-order statistics information. Therefore, an easy way to solve this problem is to replace the expectation values by the instantaneous single sample estimate based on the input vector $\mathbf{x}[n]$ and the reference signal $d[n]$, a method called stochastic gradient [22].

Finally, the update equation of the well-known LMS algorithm is expressed as

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu_0 \mathbf{x}[n] e^*[n]. \tag{2.31}$$

The convergence and stability of the LMS algorithm rely on appropriate choice of the step size μ_0 [21, 39]. A large step size leads to a fast convergence speed but also a large excess MSE at the steady state since a large value makes the algorithm hard to reach and stay at the exact minimum of the cost function. However, if the μ_0 is small, the excess mean square error is small, but the convergence speed accordingly becomes very slow. Therefore, there is a trade-off existing between convergence speed and steady state mean square error. As a result of it, we have to make a compromise to choose the appropriate step size [22].

In order to reach the steady state, the step size μ should satisfy

$$0 < \mu_0 < \frac{2}{\lambda_{max}}, \tag{2.32}$$

where λ_{max} is the maximum eigenvalue of the covariance matrix $\mathbf{R}_{xx}$. As λ_{max} has an upper limit, we can derive the following result

$$\mu_0 < \frac{2}{L\sigma_{xx}^2}, \tag{2.33}$$

where σ_{xx}^2 is the variance of input signal.

2.4 Quaternion-valued Gradient Operator and the Corresponding LMS Algorithm

As mentioned, the hypercomplex concepts such as quaternion have been introduced to solve problems related to three or four-dimensional signals. Therefore, in many of the cases, it is necessary to extend the conventional complex-valued adaptive filtering algorithms to the quaternion domain to derive the corresponding adaptive algorithms to cater for the multi-dimensional signals, such as the QLMS algorithm which has been employed in 3-D wind velocity with three orthogonal directions.

Initially, we used the existing definition of quaternion-valued gradient operation and the derived QLMS algorithm in our study. However, after detailed study, we found that the existing gradient operator definition and the derived QLMS algorithms were not consistent, mainly due to the non-commutativity rule was not strictly enforced. It appears that the authors in [6, 42, 43] used the traditional product rules as well as chain rules to derive the quaternion-valued algorithms, which are valid only if non-commutativity is not strictly enforced, while the three dimensional quaternion model we used in our research is strictly non-commutative.

We can take the derivation process of the QLMS algorithm in [6] as an example. In the LMS algorithm, the cost function is defined as $J_0(n) = e(n)e^*(n)$. Similarly, the same real-valued quadratic cost function as in LMS is employed in the quaternion domain and given by

$$J_1(n) = e(n)e^*(n) = e_a^2 + e_b^2 + e_c^2 + e_d^2, \tag{2.34}$$

where the quaternion variable $e(n)$ is the error between desired signal and the output with $e(n) = d(n) - \mathbf{w}^T(n)\mathbf{x}(n)$. $d(n)$, $\mathbf{w}(n)$, $\mathbf{x}(n)$ denote respectively the desired signal, the adaptive weight vector, and the filter input. $(\cdot)^T$, $(\cdot)^H$, and $(\cdot)^*$ represent respectively the transpose, Hermitian transpose, and quaternion conjugate operator.

With the cost function in Eq. (2.34), the following gradients need to be calculated using the stochastic gradient optimization

$$\begin{aligned} \bigtriangledown_{\mathbf{w}}(e(n)e^*(n)) &= \bigtriangledown_{\mathbf{w}_a}(e(n)e^*(n)) + \bigtriangledown_{\mathbf{w}_b}(e(n)e^*(n))i \\ &+ \bigtriangledown_{\mathbf{w}_c}(e(n)e^*(n))j + \bigtriangledown_{\mathbf{w}_d}(e(n)e^*(n))k \end{aligned} \tag{2.35}$$

where $\mathbf{w} = \mathbf{w}_a + \mathbf{w}_b i + \mathbf{w}_c j + \mathbf{w}_d k$.

For the right side of above equation, they can be expressed as

$$\begin{aligned} \bigtriangledown_{\mathbf{w}_a}(e(n)e^*(n)) &= e(n) \bigtriangledown_{\mathbf{w}_a} (e^*(n)) + \bigtriangledown_{\mathbf{w}_a}(e(n))e^*(n) \\ \bigtriangledown_{\mathbf{w}_b}(e(n)e^*(n)) &= e(n) \bigtriangledown_{\mathbf{w}_b} (e^*(n)) + \bigtriangledown_{\mathbf{w}_b}(e(n))e^*(n) \\ \bigtriangledown_{\mathbf{w}_c}(e(n)e^*(n)) &= e(n) \bigtriangledown_{\mathbf{w}_c} (e^*(n)) + \bigtriangledown_{\mathbf{w}_c}(e(n))e^*(n) \\ \bigtriangledown_{\mathbf{w}_d}(e(n)e^*(n)) &= e(n) \bigtriangledown_{\mathbf{w}_d} (e^*(n)) + \bigtriangledown_{\mathbf{w}_d}(e(n))e^*(n). \end{aligned} \tag{2.36}$$

Therefore, in order to calculate the update equation of the adaptive weight vector, the gradients shown above should be derived. In this process, the formulation $y(n) = \mathbf{w}^T(n)\mathbf{x}(n)$ and the error expression $e(n) = d(n) - \mathbf{w}^T(n)\mathbf{x}(n)$

have been employed. The full derivation of the stochastic gradient update within the QLMS algorithm is given as follows.

To calculate the derivative of the error $e(n)$ and its conjugate with respect to the weight vector $\mathbf{w}(n)$, the terms $\mathbf{w}^T(n)\mathbf{x}(n)$ and $\mathbf{x}^H(n)\mathbf{w}^*(n)$ appearing in the calculations, can be expanded as:

$$\mathbf{w}^T(n)\mathbf{x}(n) = \begin{bmatrix} \mathbf{w}_a^T\mathbf{x}_a - \mathbf{w}_b^T\mathbf{x}_b - \mathbf{w}_c^T\mathbf{x}_c - \mathbf{w}_d^T\mathbf{x}_d \\ \mathbf{w}_a^T\mathbf{x}_b + \mathbf{w}_b^T\mathbf{x}_a + \mathbf{w}_c^T\mathbf{x}_d - \mathbf{w}_d^T\mathbf{x}_c \\ \mathbf{w}_a^T\mathbf{x}_c + \mathbf{w}_c^T\mathbf{x}_a + \mathbf{w}_d^T\mathbf{x}_b - \mathbf{w}_b^T\mathbf{x}_d \\ \mathbf{w}_a^T\mathbf{x}_d + \mathbf{w}_d^T\mathbf{x}_a + \mathbf{w}_b^T\mathbf{x}_c - \mathbf{w}_c^T\mathbf{x}_b \end{bmatrix} \tag{2.37}$$

$$\mathbf{x}^H(n)\mathbf{w}^*(n) = \begin{bmatrix} \mathbf{w}_a^T\mathbf{x}_a - \mathbf{w}_b^T\mathbf{x}_b - \mathbf{w}_c^T\mathbf{x}_c - \mathbf{w}_d^T\mathbf{x}_d \\ -\mathbf{w}_a^T\mathbf{x}_b - \mathbf{w}_b^T\mathbf{x}_a - \mathbf{w}_c^T\mathbf{x}_d + \mathbf{w}_d^T\mathbf{x}_c \\ -\mathbf{w}_a^T\mathbf{x}_c - \mathbf{w}_c^T\mathbf{x}_a - \mathbf{w}_d^T\mathbf{x}_b + \mathbf{w}_b^T\mathbf{x}_d \\ -\mathbf{w}_a^T\mathbf{x}_d - \mathbf{w}_d^T\mathbf{x}_a - \mathbf{w}_b^T\mathbf{x}_c + \mathbf{w}_c^T\mathbf{x}_b \end{bmatrix} \tag{2.38}$$

Based on the Eq. (2.37) and Eq. (2.38), the derivatives from Eq. (2.36) can be reformulated into:

$$\begin{aligned} \bigtriangledown_{\mathbf{w}_a}(e(n)e^*(n)) &= e(n)(-\mathbf{x}^*(n)) + (-\mathbf{x}(n))e^*(n) \\ &= -e(n)\mathbf{x}^*(n) - \mathbf{x}(n)e^*(n) \end{aligned} \tag{2.39}$$

$$\begin{aligned} \bigtriangledown_{\mathbf{w}_b}(e(n)e^*(n))i &= e(n)(\mathbf{x}_b + \mathbf{x}_a i - \mathbf{x}_d j + \mathbf{x}_c k)i \\ &\quad +(\mathbf{x}_b - \mathbf{x}_a i + \mathbf{x}_d j - \mathbf{x}_c k)e^*(n)i \\ &= e(n)(-\mathbf{x}_a + \mathbf{x}_b i + \mathbf{x}_c j + \mathbf{x}_d k) \\ &\quad +(\mathbf{x}_a + \mathbf{x}_b i - \mathbf{x}_c j - \mathbf{x}_d k)e^*(n) \end{aligned} \tag{2.40}$$

$$\begin{aligned}\bigtriangledown_{\mathbf{w}_c}(e(n)e^*(n))j &= e(n)(\mathbf{x}_c+\mathbf{x}_d i+\mathbf{x}_a j-\mathbf{x}_b k)j\\ &\quad +(\mathbf{x}_c-\mathbf{x}_d i-\mathbf{x}_a j+\mathbf{x}_b k)e^*(n)j\\ &= e(n)(-\mathbf{x}_a+\mathbf{x}_b i+\mathbf{x}_c j+\mathbf{x}_d k)\\ &\quad +(\mathbf{x}_a-\mathbf{x}_b i+\mathbf{x}_c j-\mathbf{x}_d k)e^*(n) \qquad (2.41)\end{aligned}$$

$$\begin{aligned}\bigtriangledown_{\mathbf{w}_d}(e(n)e^*(n))k &= e(n)(\mathbf{x}_d-\mathbf{x}_c i+\mathbf{x}_b j+\mathbf{x}_a k)k\\ &\quad +(\mathbf{x}_d+\mathbf{x}_c i-\mathbf{x}_b j-\mathbf{x}_a k)e^*(n)k\\ &= e(n)(-\mathbf{x}_a+\mathbf{x}_b i+\mathbf{x}_c j+\mathbf{x}_d k)\\ &\quad +(\mathbf{x}_a-\mathbf{x}_b i-\mathbf{x}_c j+\mathbf{x}_d k)e^*(n). \qquad (2.42)\end{aligned}$$

Substituting Eq. (2.39) - Eq. (2.42) into Eq. (2.35), we obtain the final expression for the gradient of the cost function Eq. (2.34) in the following form as

$$\begin{aligned}\bigtriangledown_{\mathbf{w}}(e(n)e^*(n)) &= \bigtriangledown_{\mathbf{w}_a}(e(n)e^*(n))+[\bigtriangledown_{\mathbf{w}_b}(e(n)e^*(n))i\\ &\quad +\bigtriangledown_{\mathbf{w}_c}(e(n)e^*(n))j+\bigtriangledown_{\mathbf{w}_d}(e(n)e^*(n))k]\\ &= -e(n)\mathbf{x}^*(n)-\mathbf{x}(n)e^*(n)\\ &\quad +[-3e(n)\mathbf{x}^*(n)+(3\mathbf{x}_a-\mathbf{x}_b i-\mathbf{x}_c j-\mathbf{x}_d k)e^*(n)]\\ &= -e(n)\mathbf{x}^*(n)-\mathbf{x}(n)e^*(n)\\ &\quad +[-3e(n)\mathbf{x}^*(n)+\mathbf{x}(n)e^*(n)+2\mathbf{x}_a e^*(n)]\\ &= -4e(n)\mathbf{x}^*(n)-\mathbf{x}(n)e^*(n)+\mathbf{x}^*(n)e^*(n)+2\mathbf{x}_a e^*(n)\\ &= -4e(n)\mathbf{x}^*(n)+2\mathbf{x}^*(n)e^*(n). \qquad (2.43)\end{aligned}$$

As the weight update is $\triangle\mathbf{w}(n)=-\mu\bigtriangledown_{\mathbf{w}}(e(n)e^*(n))$ with step size μ_1, consequently, the update of the adaptive weight vector of QLMS can be expressed as:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu_1(2e(n)\mathbf{x}^*(n) - \mathbf{x}^*(n)e^*(n)) \tag{2.44}$$

Since the authors did not state explicitly when and how the non-commutativity condition should be enforced, it is difficult to apply the HR derivative in a general case. As a result, we had to re-define the gradient operation and derive a new version of the QLMS algorithm, which admits the general product rules and chain rules.

The HC-derivative in [44] presents the same complication. Besides, the C-derivative is simpler than the H-derivative only when it is applied to complex-valued or real-valued functions. It no longer possesses the simplicity when it is applied to quaternion-valued functions. For example, the product rule is no longer valid even for the C-derivative if non-commutativity is enforced.

The GHR derivative in [45], which is proposed independent of our restricted HR, is similar to our restricted HR derivative. It reverses to our definition when μ is chosen among $(1, i, j, k)$. However, for the restricted HR, we are able to prove some powerful general results. It is not obvious that these results are also true in GHR calculus.

2.5 Quaternion-valued Discrete Fourier Transform

Fourier transform has been used extensively in signal processing ever since the discovery of the Fast Fourier Transform (FFT). On the other hand, since the hypercomplex number, particularly, the quaternion, was introduced to deal with multi-dimensional signals, quaternion-valued Fourier transform was presented in [25, 46, 47] to represent 3-D pixels for color image processing. The first application of quaternion-valued Fourier transform to color images was studied by Sangwine in [48] using a discrete version of Ells transform. Then the quaternion-valued discrete Fourier transform (QDFT) was introduced by [26].

For QDFT [26, 49], there are two different types, which are left-side and right-side, respectively, due to the non-commutativity property of the quaternion [50]. Take the right-side QDFT as an example first. As a quaternion has three imaginary components, it is essential to specify the transform axis. With the i transform axis, the QDFT of the input data vector $\mathbf{x}[n] = [x[n], x[n-1], \cdots, x[n-L+1]]^T$ with a length of L can be expressed as

$$X_k[n] = \sum_{l=0}^{L} x[n-l]e^{-i2\pi kl/L}, \tag{2.45}$$

where $k, l = 1, 2, \cdots, L$. All of the L components $X_k[n]$ form the vector $\mathbf{X}[n] = \begin{bmatrix} X_1 & ... & X_L \end{bmatrix}^T$.

The inverse QDFT (IQDFT) of $\mathbf{X}[n]$ is given by [51]

$$x[n-l] = \frac{1}{L}\sum_{k=1}^{L} X_k e^{i2\pi kl/L}. \tag{2.46}$$

With a simplified notation, the output of the L-point QDFT is

$$\mathbf{X}[n] = \text{QDFT}[\mathbf{x}[n]] \tag{2.47}$$

and its inverse form is

$$\mathbf{x}[n] = \text{IQDFT}[\mathbf{X}[n]]. \tag{2.48}$$

While for the left-side QDFT, the transform axis should be multiplied by the left-side rather than right. For example, the left-side QDFT with the j transform axis of input data vector $\mathbf{x}[n]$ is given by

$$X_m[n] = \sum_{l=1}^{L} e^{-j2\pi ml/L} x[n-l], \tag{2.49}$$

where $m = 1, 2, \cdots, L$. Therefore, the inverse QDFT (IQDFT) of $\mathbf{X}_m$ is

$$x[n-l] = \frac{1}{L} \sum_{m=1}^{L} e^{j2\pi ml/L} X_m. \tag{2.50}$$

Similarly, the transform axis k has the same format.

2.6 Summary

In summary, the algebra of quaternion including the involutions and QDFT as well as a brief review of previous research for both real-valued and quaternion-valued gradient operator and the derived adaptive algorithms has been presented. Although there have been some derivations about the quaternion-valued gradient operator with different level of details, it is still not completely clear how this operator can be derived in the most general case and how it can be applied to various signal processing problems. In the following chapters, we will provide a general quaternion-valued gradient operator and then derive a series of quaternion-valued adaptive algorithms, which will be applied to different areas, including adaptive beamforming and wind profile prediction.

Chapter 3

General Quaternion-valued Gradient Operation and Its Applications

As mentioned above, quaternions are an extension of complex numbers from 2D plane to the 3D or 4D spaces and form one of the four existing division algebra (real R, complex C, quaternion H and octonions O). A Hermitian Real-valued (HR) gradient operator was proposed in [42] and the interesting formulation appears to provide a general and flexible framework that could potentially have wide applications. However, it has only been applied to real-valued functions and linear quaternion-valued functions. In order to consider more general quaternion-valued functions, we propose a pair of restricted HR gradient operators, the left and the right restricted HR gradient operators, based on the previous work on the HR gradient operator in [42] and our recent work in [10].

Based on the general quaternion-valued gradient operator, we will derive the standard QLMS algorithm in a new way and also move to the nonlinear adaptive filtering area [52], by developing a quaternion-valued nonlinear adaptive filtering algorithm. Moreover, there is a class of algorithms that specifically designed for sparse system identification, where the unknown system only has a few large coefficients while the remaining ones own a very small amplitude that they can be ignored without significant effect on the overall performance of the system. To improve the performance of the QLMS algorithm for sparse system identification, the ZA-QLMS algorithm is derived in this chapter.

This chapter is organised as follows. The restricted HR gradient operator and the right restricted HR gradient operator are developed in Sections 3.1 and 3.2, with their properties and rules introduced in Section 3.3. Explicit expressions for the derivatives for a wide range of functions are provided in Section 3.4 and results for the right restricted HR operator are summarised in Section 3.5. The increment of a general quaternion function and applications of the general gradient operator to quaternion-valued adaptive algorithms are presented in Section 3.6. The ZA-QLMS algorithm for sparse system identification is given in Section 3.7, followed by a summary in Section 3.8.

3.1 The restricted HR gradient operator

Let f be a quaternion-valued function of a non-commutative quaternion q. We use the notation

$$f(q) = f_a + f_b i + f_c j + f_d k, \tag{3.1}$$

where $f_a, ..., f_d$ are the components of f. f can also be viewed as a function of the four components of q, i.e., $f = f(q_a, q_b, q_c, q_d)$. In this view f is a quaternion-valued function on R^4.

A so-called HR gradient of $f(q)$ was introduced in [42], which has been applied to real-valued functions and linear quaternion-valued functions. In order to find the gradients of more general quaternion-valued functions, we follow a similar approach to propose a 'restricted' HR gradient operator (some of the derivation was first presented in [10]). To motivate the definitions, we consider the differential $df(q)$ with respect to differential

$$dq := dq_a + dq_b i + dq_c j + dq_d k. \tag{3.2}$$

We observe that

$$df = df_a + idf_b + jdf_c + kdf_d, \tag{3.3}$$

where

$$df_a = \frac{\partial f_a}{\partial q_a}dq_a + \frac{\partial f_a}{\partial q_b}dq_b + \frac{\partial f_a}{\partial q_c}dq_c + \frac{\partial f_a}{\partial q_d}dq_d. \tag{3.4}$$

According to (2.11), we have

$$dq_a = (dq + dq^i + dq^j + dq^k)/4. \tag{3.5}$$

Making use of this and similar expressions for dq_b, dq_c and dq_d, we find an expression for df_a in terms of the differentials dq, dq^i, dq^j and dq^k. More details are given below.

We consider $df = df_a + idf_b + jdf_c + kdf_d$. By definition and follow (3.4), we have

$$df_b = \frac{\partial f_b}{\partial q_a}dq_a + \frac{\partial f_b}{\partial q_b}dq_b + \frac{\partial f_b}{\partial q_c}dq_c + \frac{\partial f_b}{\partial q_d}dq_d, \tag{3.6}$$

$$df_c = \frac{\partial f_c}{\partial q_a}dq_a + \frac{\partial f_c}{\partial q_b}dq_b + \frac{\partial f_c}{\partial q_c}dq_c + \frac{\partial f_c}{\partial q_d}dq_d, \tag{3.7}$$

$$df_d = \frac{\partial f_d}{\partial q_a}dq_a + \frac{\partial f_d}{\partial q_b}dq_b + \frac{\partial f_d}{\partial q_c}dq_c + \frac{\partial f_d}{\partial q_d}dq_d. \tag{3.8}$$

Using the relations

$$dq_a = \frac{1}{4}(dq + dq^i + dq^j + dq^k), \tag{3.9}$$

$$dq_b = \frac{1}{4i}(dq + dq^i - dq^j - dq^k), \tag{3.10}$$

$$dq_c = \frac{1}{4j}(dq - dq^i + dq^j - dq^k), \tag{3.11}$$

$$dq_d = \frac{1}{4k}(dq - dq^i - dq^j + dq^k), \tag{3.12}$$

we may rewrite df_γ with $\gamma \in \{a, b, c, d\}$ as follows

$$\begin{aligned} df_\gamma = \quad & \frac{1}{4}(\frac{\partial f_\gamma}{\partial q_a} - i\frac{\partial f_\gamma}{\partial q_b} - j\frac{\partial f_\gamma}{\partial q_c} - k\frac{\partial f_\gamma}{\partial q_d})dq \\ & + \frac{1}{4}(\frac{\partial f_\gamma}{\partial q_a} - i\frac{\partial f_\gamma}{\partial q_b} + j\frac{\partial f_\gamma}{\partial q_c} + k\frac{\partial f_\gamma}{\partial q_d})dq^i \\ & + \frac{1}{4}(\frac{\partial f_\gamma}{\partial q_a} + i\frac{\partial f_\gamma}{\partial q_b} - j\frac{\partial f_\gamma}{\partial q_c} + k\frac{\partial f_\gamma}{\partial q_d})dq^j \\ & + \frac{1}{4}(\frac{\partial f_\gamma}{\partial q_a} + i\frac{\partial f_\gamma}{\partial q_b} + j\frac{\partial f_\gamma}{\partial q_c} - k\frac{\partial f_\gamma}{\partial q_d})dq^k \end{aligned}$$

which can be written as

$$df_\gamma = \frac{1}{4}\sum_\nu \left(\sum_{(\phi,\mu)} \frac{\partial f_\gamma}{\partial q_\phi}\mu^\nu \right) dq^\nu \tag{3.13}$$

where $(\phi, \mu) \in \{(a, 1), (b, -i), (c, -j), (d, -k)\}$, $\nu \in \{1, i, j, k\}$, and μ^ν is the

ν-involution of μ. Therefore

$$\begin{aligned} df &= df_a + idf_b + jdf_c + kdf_d \\ &= \frac{1}{4}\sum_{\nu}\left(\sum_{(\phi,\mu)}\frac{\partial(f_a + if_b + jf_c + kf_d)}{\partial q_\phi}\mu^\nu\right)dq^\nu \\ &= \frac{1}{4}\sum_{\nu}\left(\sum_{(\phi,\mu)}\frac{\partial f}{\partial q_\phi}\mu^\nu\right)dq^\nu. \end{aligned} \tag{3.14}$$

Note that, because μ^ν and dq^ν are quaternions, to obtain the last equation, we need to multiply df_b, df_c and df_d by i, j, and k from the left.

We finally arrive at

$$df = Ddq + D_i dq^i + D_j dq^j + D_k dq^k\ , \quad \text{where} \tag{3.15}$$

$$D := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} - \frac{\partial f}{\partial q_b}i - \frac{\partial f}{\partial q_c}j - \frac{\partial f}{\partial q_d}k\right), \tag{3.16}$$

$$D_i := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} - \frac{\partial f}{\partial q_b}i + \frac{\partial f}{\partial q_c}j + \frac{\partial f}{\partial q_d}k\right), \tag{3.17}$$

$$D_j := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + \frac{\partial f}{\partial q_b}i - \frac{\partial f}{\partial q_c}j + \frac{\partial f}{\partial q_d}k\right), \tag{3.18}$$

$$D_k := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + \frac{\partial f}{\partial q_b}i + \frac{\partial f}{\partial q_c}j - \frac{\partial f}{\partial q_d}k\right). \tag{3.19}$$

Thus one may define the partial derivatives of $f(q)$ as follows:

$$\frac{\partial f}{\partial q} := D, \quad \frac{\partial f}{\partial q^i} := D_i, \frac{\partial f}{\partial q^j} := D_j, \frac{\partial f}{\partial q^k} := D_k. \tag{3.20}$$

Introducing operators

$$\nabla_q := (\partial/\partial q, \partial/\partial q^i, \partial/\partial q^j, \partial/\partial q^k), \tag{3.21}$$

and

$$\nabla_r := (\partial/\partial q_a, \partial/\partial q_b, \partial/\partial q_c, \partial/\partial q_d), \tag{3.22}$$

equations (3.16-3.20) may be written as

$$\nabla_q f = \nabla_r f J^H \tag{3.23}$$

where the Jacobian matrix

$$J = \frac{1}{4}\begin{bmatrix} 1 & i & j & k \\ 1 & i & -j & -k \\ 1 & -i & j & -k \\ 1 & -i & -j & k \end{bmatrix} \tag{3.24}$$

and J^H is the Hermitian transpose of J [42]. Using $JJ^H = J^HJ = 1/4$, we may also write

$$\nabla_q f J = \frac{1}{4}\nabla_r f, \tag{3.25}$$

which is the inverse formulae for the derivatives.

We call the gradient operator defined by (3.23) the restricted HR gradient operator. The operator is closely related to the HR operator introduced in [42]. However, in the original definition of the HR operator, the Jacobian J appears on the left-hand side of $\nabla_r f$, whereas in our definition it appears on the right (as the Hermitian transpose).

The differential df is related to $\nabla_q f$ by

$$df = \frac{\partial f}{\partial q}dq + \frac{\partial f}{\partial q^i}dq^i + \frac{\partial f}{\partial q^j}dq^j + \frac{\partial f}{\partial q^k}dq^k. \tag{3.26}$$

Due to the non-commutativity of quaternion products, the order of the factors in the products of the above equation (as well as equations (3.16-3.19)) can not be swapped. In fact, one may call the above operator the left restricted HR gradient operator.

3.2 The right restricted HR gradient operator

On the other hand, we notice that the prefactors in (3.10-3.12) may be moved to the right-hand side of the other factors, i.e., we may write

$$dq_a = (dq + dq^i + dq^j + dq^k)\frac{1}{4}, \tag{3.27}$$

$$dq_b = (dq + dq^i - dq^j - dq^k)\frac{1}{4i}, \tag{3.28}$$

$$dq_c = (dq - dq^i + dq^j - dq^k)\frac{1}{4j}, \tag{3.29}$$

$$dq_d = (dq - dq^i - dq^j + dq^k)\frac{1}{4k}. \tag{3.30}$$

Using these relations, we may find another expression for df_γ following the procedure above:

$$df_\gamma = \frac{1}{4}\sum_\nu dq^\nu \left(\sum_{(\phi,\mu)} \mu^\nu \frac{\partial f_\gamma}{\partial q_\phi}\right). \tag{3.31}$$

The expression is different from (3.13), in that the differentials dq^ν are on the left of μ^ν. Therefore, we derive

$$\begin{aligned} df &= df_a + df_b i + df_c j + df_d k \\ &= \frac{1}{4}\sum_\nu dq^\nu \left(\sum_{(\phi,\mu)} \mu^\nu \frac{\partial(f_a + f_b i + f_c j + f_d k)}{\partial q_\phi}\right) \\ &= \frac{1}{4}\sum_\nu dq^\nu \left(\sum_{(\phi,\mu)} \mu^\nu \frac{\partial f}{\partial q_\phi}\right), \end{aligned} \tag{3.32}$$

which is the basis for the definitions for the right restricted HR derivatives. One may define the right restricted HR gradient operator by

$$(\nabla_q^R f)^T := J^*(\nabla_r f)^T, \text{ where} \tag{3.33}$$

$$\nabla_q^R := (\partial^R/\partial q, \partial^R/\partial q^i, \partial^R/\partial q^j, \partial^R/\partial q^k),$$

and

$$\frac{\partial^R f}{\partial q} := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} - i\frac{\partial f}{\partial q_b} - j\frac{\partial f}{\partial q_c} - k\frac{\partial f}{\partial q_d}\right), \tag{3.34}$$

$$\frac{\partial^R f}{\partial q^i} := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} - i\frac{\partial f}{\partial q_b} + j\frac{\partial f}{\partial q_c} + k\frac{\partial f}{\partial q_d}\right), \tag{3.35}$$

$$\frac{\partial^R f}{\partial q^j} := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + i\frac{\partial f}{\partial q_b} - j\frac{\partial f}{\partial q_c} + k\frac{\partial f}{\partial q_d}\right), \tag{3.36}$$

$$\frac{\partial^R f}{\partial q^k} := \frac{1}{4}\left(\frac{\partial f}{\partial q_a} + i\frac{\partial f}{\partial q_b} + j\frac{\partial f}{\partial q_c} - k\frac{\partial f}{\partial q_d}\right). \tag{3.37}$$

The right restricted HR gradient operator is related to the differential df by

$$df = dq\frac{\partial^R f}{\partial q} + dq^i\frac{\partial^R f}{\partial q^i} + dq^j\frac{\partial^R f}{\partial q^j} + dq^k\frac{\partial^R f}{\partial q^k}. \tag{3.38}$$

In general, the left and right restricted HR gradients are not the same. For example, even for the simplest linear function $f(q) = q_0 q$ with $q_0 \in H$ a constant, we have

$$\frac{\partial q_0 q}{\partial q} = q_0, \ \frac{\partial^R q_0 q}{\partial q} = R(q_0). \tag{3.39}$$

However, we will show later that the two gradients coincide for a class of functions. In particular, they are the same for real-valued quaternion functions. The relation between the gradients and the differential is an important ingredient of gradient-based methods, which we will discuss further later.

3.3 Properties and Rules of the Operator

We will now focus on the left restricted HR gradient and simply call it the restricted HR gradient unless stated otherwise. It can be easily calculated from the definitions, that

$$\frac{\partial q}{\partial q} = 1,\ \frac{\partial q^{\nu}}{\partial q} = 0,\ \frac{\partial q^{*}}{\partial q} = -\frac{1}{2}, \tag{3.40}$$

where $\nu \in \{i, j, k\}$. However, in order to find the derivatives for more complex quaternion functions, it is useful to first establish the rules of the gradient operators. We will see that some of the usual rules do not apply due to the non-commutativity of quaternion products.

1. Left-linearity: for arbitrary constant quaternions α and β, and functions $f(q)$ and $g(q)$, we have

$$\frac{\partial(\alpha f + \beta g)}{\partial q^{\nu}} = \alpha\frac{\partial f}{\partial q^{\nu}} + \beta\frac{\partial g}{\partial q^{\nu}} \tag{3.41}$$

for $\nu \in \{1, i, j, k\}$ with $q^1 := q$. However, linearity does not hold for right multiplications, i.e., in general

$$\frac{\partial f\alpha}{\partial q} \neq \frac{\partial f}{\partial q}\alpha. \tag{3.42}$$

This is because, according to the definition (3.16),

$$\frac{\partial f\alpha}{\partial q} = \frac{1}{4}\sum_{(\phi,\gamma)}\frac{\partial f}{\partial q_{\phi}}\alpha\gamma \tag{3.43}$$

for $(\phi, \gamma) \in \{(a, 1), (b, -i), (c, -j), (d, -k)\}$. However, $\alpha\gamma \neq \gamma\alpha$ in general. Therefore it is different from $(\partial f/\partial q)\alpha$, which is

$$\frac{1}{4}\left(\frac{\partial f}{\partial q_a} - \frac{\partial f}{\partial q_b}i - \frac{\partial f}{\partial q_c}j - \frac{\partial f}{\partial q_d}k\right)\alpha. \tag{3.44}$$

2. The first product rule: the following product rule holds:

$$\nabla_q(fg) = f\nabla_q g + [(\nabla_r f)g]J^H. \tag{3.45}$$

For example,

$$\frac{\partial f q}{\partial q} = f\frac{\partial g}{\partial q} + \frac{1}{4}\left(\frac{\partial f}{\partial q_a} g - \frac{\partial f}{\partial q_b} gi - \frac{\partial f}{\partial q_c} gj - \frac{\partial f}{\partial q_d} gk\right). \tag{3.46}$$

Thus the product rule in general is different from the usual one.

3. The second product rule: However, the usual product rule applies to differentiation with respect to real variables, i.e.,

$$\frac{\partial f g}{\partial q_\phi} = \frac{\partial f}{\partial q_\phi} g + f\frac{\partial g}{\partial q_\phi} \tag{3.47}$$

for $\phi = a, b, c$, or d.

4. The third product rule: The usual product rule also applies if at least one of the two functions $f(q)$ and $g(q)$ is real-valued, i.e.,

$$\frac{\partial f q}{\partial q} = f\frac{\partial g}{\partial q} + \frac{\partial f}{\partial q} g. \tag{3.48}$$

5. The first chain rule: For a composite function $f(g(q))$, $g(q) := g_a + g_b i + g_c j + g_d k$ being a quaternion-valued function, we have the following chain rule:

$$\nabla_q f = (\nabla_q^g f) M \tag{3.49}$$

where $\nabla_q^g := (\partial/\partial g, \partial/\partial g^i, \partial/\partial g^j, \partial/\partial g^k)$ and M is a 4×4 matrix with element $M_{\mu\nu} = \partial g^\mu / \partial q^\nu$ for $\mu, \nu \in \{1, i, j, k\}$ and $g^\mu = -\mu g \mu$ (g^1 is understood as the same as g). Explicitly, we may write

$$\frac{\partial f}{\partial q^\nu} = \sum_\mu \frac{\partial f}{\partial g^\mu}\frac{\partial g^\mu}{\partial q^\nu}. \tag{3.50}$$

The proof is outlined below.

The function $f(g(q))$ may be view as a function of intermediate variables g_a, g_b, g_c and g_d. Using the usual chain rule, we have

$$\begin{aligned} \frac{\partial f}{\partial q_\beta} &= \frac{\partial f}{\partial g_a}\frac{\partial g_a}{\partial q_\beta} + \frac{\partial f}{\partial g_b}\frac{\partial g_b}{\partial q_\beta} + \frac{\partial f}{\partial g_c}\frac{\partial g_c}{\partial q_\beta} + \frac{\partial f}{\partial g_d}\frac{\partial g_d}{\partial q_\beta} \\ &= \sum_\phi \frac{\partial f}{\partial g_\phi}\frac{\partial g_\phi}{\partial q_\beta}, \end{aligned} \tag{3.51}$$

with $\beta, \phi \in \{a, b, c, d\}$, which gives $\nabla_r f = (\nabla_r^g f)P$, where P is a 4×4 matrix with $P_{\phi\beta} = \partial g_\phi / \partial q_\beta$. With $(\nabla_r f)J^H = \nabla_q f$, and $\nabla_r^g f = 4(\nabla_q^g f)J$, the above equation leads to

$$\nabla_q f = 4(\nabla_q^g f)JPJ^H, \tag{3.52}$$

where it is easy to show that $4JPJ^H = M$.

6. The second chain rule: The above chain rule uses g and its involutions as the intermediate variables. It is sometimes convenient to use the real components of g for that purpose instead. In this case, the following chain rule may be used:

$$\nabla_q f = (\nabla_r^g f)O \tag{3.53}$$

where O is a 4×4 matrix with entry $O_{\phi\nu} = \partial g_\phi / \partial q^\nu$ with $\phi \in \{a, b, c, d\}$ and $\nu \in \{1, i, j, k\}$, and $\nabla_r^g := (\partial/\partial g_a, \partial/\partial g_b, \partial/\partial g_c, \partial/\partial g_d)$. Explicitly, we have

$$\frac{\partial f}{\partial q^\nu} = \sum_\phi \frac{\partial f}{\partial g_\phi}\frac{\partial g_\phi}{\partial q^\nu}. \tag{3.54}$$

7. The third chain rule: if the intermediate function $g(q)$ is real-valued, i.e., $g = g_a$, then from the second chain rule, we obtain

$$\frac{\partial f}{\partial q^\nu} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial q^\nu}. \tag{3.55}$$

8. $f(q)$ is not independent of q^i, q^j or q^k in the sense that, in general,

$$\frac{\partial f(q)}{\partial q^i} \neq 0, \frac{\partial f(q)}{\partial q^j} \neq 0, \frac{\partial f(q)}{\partial q^k} \neq 0. \tag{3.56}$$

This can be illustrated by $f(q) = q^2$. Using the first product rule (equation (3.45)), we have

$$\frac{\partial q^2}{\partial q^i} = q\frac{\partial q}{\partial q^i} + \frac{1}{4}\sum_{(\phi,\nu)} \frac{\partial q}{\partial q_\phi} q\nu$$

for $(\phi, \nu) \in \{(a, 1), (b, i), (c, -j), (d, -k)\}$. It can then be shown that

$$\frac{\partial q^2}{\partial q^i} = q_b i, \ \frac{\partial q^2}{\partial q^j} = q_c j, \ \frac{\partial q^2}{\partial q^k} = q_d k. \tag{3.57}$$

This property demonstrates the intriguing difference between the HR derivative and the usual derivatives, although we can indeed show that

$$\frac{\partial q}{\partial q^\nu} = 0. \tag{3.58}$$

One implication of this observation is that, for a nonlinear algorithm involving simultaneously more than one gradients $\partial f / \partial q^\nu$, we have to take care to include all the terms.

3.4 Restricted HR Derivatives for a Class of Regular Functions

Using the above operation rules, we may find explicit expressions for the derivatives for a whole range of functions. We first introduce the following lemma:

Lemma 1. The derivative of the power function $f(q) = (q - q_0)^n$, with integer n and constant quaternion q_0, is

$$\frac{\partial f(q)}{\partial q} = \frac{1}{2}\left(n\tilde{q}^{n-1} + \frac{\tilde{q}^n - \tilde{q}^{*n}}{\tilde{q} - \tilde{q}^*}\right), \tag{3.59}$$

with $\tilde{q} = q - q_0$.

Remark. The division in $(\tilde{q}^n - \tilde{q}^{*n})/(\tilde{q} - \tilde{q}^*)$ is understood as $(\tilde{q}^n - \tilde{q}^{*n})(\tilde{q} - \tilde{q}^*)^{-1}$ or $(\tilde{q} - \tilde{q}^*)^{-1}(\tilde{q}^n - \tilde{q}^{*n})$ which are the same since the two factors commute. The division operations in what follows are understood in the same way.

Proof. The lemma is obviously true for $n = 0$. Let $n \geq 1$, we apply the first product rule, and find

$$\frac{\partial (q - q_0)^n}{\partial q} = \tilde{q}\frac{\partial \tilde{q}^{n-1}}{\partial q} + R(\tilde{q}^{n-1}) \tag{3.60}$$

where $R(\tilde{q}^{n-1})$ is the real part of $\tilde{q}^{n-1}$. We then obtain by induction

$$\frac{\partial (q - q_0)^n}{\partial q} = \sum_{m=0}^{n-1} \tilde{q}^m R(\tilde{q}^{n-1-m}). \tag{3.61}$$

Using $R(\tilde{q}^{n-1-m}) = \frac{1}{2}(\tilde{q}^{n-1-m} + \tilde{q}^{*(n-1-m)})$, the summations can be evaluated explicitly, which is given by

$$\begin{aligned}
\frac{\partial (q - q_0)^n}{\partial q} &= \sum_{m=0}^{n-1} \tilde{q}^m R(\tilde{q}^{n-1-m}) \\
&= \frac{1}{2}\left(\sum_{m=0}^{n-1} (\tilde{q}^{n-1-m} + \tilde{q}^{*(n-1-m)})\tilde{q}^m\right) \\
&= \frac{1}{2}\left(\sum_{m=0}^{n-1} \tilde{q}^{n-1} + \sum_{m=0}^{n-1} \tilde{q}^{*(n-1)} \left(\frac{\tilde{q}^m}{\tilde{q}^{*m}}\right)^m\right) \\
&= \frac{1}{2}\left(n\tilde{q}^{n-1} + \frac{\tilde{q}^n - \tilde{q}^{*n}}{\tilde{q} - \tilde{q}^*}\right).
\end{aligned} \tag{3.62}$$

For $n < 0$, we use the recurrent relation

$$\frac{\partial ((q - q_0)^{-n})}{\partial q} = \tilde{q}^{-1}\left[\frac{\partial \tilde{q}^{-(n-1)}}{\partial q} - R(\tilde{q}^{-n})\right] \tag{3.63}$$

and the result

$$\frac{\partial (q-q_0)^{-1}}{\partial q} = -\tilde{q}^{-1} R(\tilde{q}^{-1}). \tag{3.64}$$

and also we have used the following relation

$$\frac{\partial q^{-1}}{\partial q} = -q^{-1} R(q^{-1}). \tag{3.65}$$

To show this result, we note $\partial (qq^{-1})/\partial q = \partial 1/\partial q = 0$. Thus

$$\begin{aligned} 0 &= q\frac{\partial q^{-1}}{\partial q} + \frac{1}{4}(q^{-1} - iq^{-1}i - jq^{-1}j - kq^{-1}k) \\ &= q\frac{\partial q^{-1}}{\partial q} + R(q^{-1}), \end{aligned} \tag{3.66}$$

from which the result follows. We have used (3.16) and the fact that

$$\frac{\partial q}{\partial q_a} = 1, \frac{\partial q}{\partial q_b} = i, \frac{\partial q}{\partial q_c} = j, \frac{\partial q}{\partial q_d} = k. \tag{3.67}$$

The proof also uses the following recurrent relation

$$\frac{\partial q^{-n}}{\partial q} = q^{-1}\left[\frac{\partial q^{-(n-1)}}{\partial q} - R(q^{-n})\right], \tag{3.68}$$

which can be shown as follows: using the first product rule, we have

$$\begin{aligned} \frac{\partial q^{-n}}{\partial q} &= q^{-1}\frac{\partial q^{-(n-1)}}{\partial q} + \frac{1}{4}\left(\frac{\partial q^{-1}}{\partial q_a} q^{-(n-1)} - \right. \\ &\left. \frac{\partial q^{-1}}{\partial q_b} q^{-(n-1)} i - \frac{\partial q^{-1}}{\partial q_c} q^{-(n-1)} j - \frac{\partial q^{-1}}{\partial q_d} q^{-(n-1)} k\right). \end{aligned} \tag{3.69}$$

Using the fact $\partial qq^{-1}/\partial q_\phi = 0$ for $\phi \in \{a, b, c, d\}$, and the second product rule, we find

$$\frac{\partial q^{-1}}{\partial q_\phi} = -q^{-1}\frac{\partial q}{\partial q_\phi} q^{-1}. \tag{3.70}$$

Thus

$$\begin{aligned}
\frac{\partial q^{-n}}{\partial q} &= q^{-1}\frac{\partial q^{-(n-1)}}{\partial q} - \frac{q^{-1}}{4}\left(q^{-n} - iq^{-n}i - jq^{-n}j - kq^{-n}k\right) \\
&= q^{-1}\frac{\partial q^{-(n-1)}}{\partial q} - q^{-1}R(q^{-n}) \\
&= q^{-1}[\frac{\partial q^{-(n-2)}}{\partial q} - R(q^{-(n-1)})] - q^{-1}R(q^{-n}) \\
&= -\sum_{m=1}^{n} q^{-m}R(q^{-(n+1-m)}) \\
&= -\sum_{m=1}^{n} q^{-(n+1-m)}R(q^{-m}) \\
&= -\frac{1}{2}\left(nq^{-(n+1)} + \frac{q^{-*}q^{-(n+1)} - q^{-*(n+1)}q^{-1}}{q^{-1} - q^{-*}}\right) \\
&= -\frac{1}{2}\left(nq^{-(n+1)} + (q^{-1} - q^{-*})^{-1}(qq^{*})^{-1}(qq^{*})(q^{-*}q^{-(n+1)} - q^{-*(n+1)}q^{-1})\right) \\
&= \frac{1}{2}\left(-nq^{-(n+1)} + \frac{q^{-n} - q^{-n*}}{q - q^{*}}\right). \qquad (3.71)
\end{aligned}$$

Equation (3.59) is proven by using induction as for $n > 0$ and using recurrent relation for $n < 0$.

Theorem 1. Assuming $f : H \to H$ admits a power series representation $f(q) := g(\tilde{q}) := \sum_{n=-\infty}^{\infty} a_n\tilde{q}^n$, with a_n being a quaternion constant and $\tilde{q} = q - q_0$, for $R_1 \leq |\tilde{q}| \leq R_2$ with $R_1, R_2 > 0$ being some constants, then

$$\frac{\partial f(q)}{\partial q} = \frac{1}{2}\left[f'(q) + (g(\tilde{q}) - g(\tilde{q}^{*}))(\tilde{q} - \tilde{q}^{*})^{-1}\right], \qquad (3.72)$$

where $f'(q)$ is the derivative in the usual sense, i.e.,

$$f'(q) := \sum_{n=-\infty}^{\infty} na_n\tilde{q}^{n-1} = \sum_{n=-\infty}^{\infty} na_n(q - q_0)^{n-1}. \qquad (3.73)$$

Proof. Using Lemma 1 and the restricted left-linearity of HR gradients, we have

$$\begin{aligned}\frac{\partial f}{\partial q} &= \frac{1}{2}\sum_{n=-\infty}^{\infty} a_n[n\tilde{q}^{n-1} + (\tilde{q}^n - \tilde{q}^{*n})(\tilde{q} - \tilde{q}^*)^{-1}] \\ &= \frac{1}{2}f'(q) + \frac{1}{2}\left[\sum_{n=\infty}^{-\infty} a_n(\tilde{q}^n - \tilde{q}^{*n})\right](\tilde{q} - \tilde{q}^*)^{-1} \\ &= \frac{1}{2}[f'(q) + (g(\tilde{q}) - g(\tilde{q}^*))(\tilde{q} - \tilde{q}^*)^{-1}],\end{aligned}$$

proving the theorem.

The functions $f(q)$ form a class of regular functions on H. A full discussion of such functions is beyond the scope of this thesis. However, we note that a similar class of functions have been discussed in [53]. A parallel development for the former is possible, and will be a topic of our future research. Meanwhile, we observe that many useful elementary functions satisfy the conditions in Theorem 1. To illustrate the application of the theorem, we list below the derivatives of a number of such functions.

Example 1. Exponential function $f(q) = e^q$ has representation

$$e^q := \sum_{n=0}^{\infty} \frac{q^n}{n!}. \tag{3.74}$$

Applying Theorem 1 with $a_n = 1/n!$ and $q_0 = 0$, we have

$$\frac{\partial e^q}{\partial q} = \frac{1}{2}\left(e^q + \frac{e^q - e^{q^*}}{q - q^*}\right). \tag{3.75}$$

The exponential function $f(q) = e^q$ is defined by the series expression $e^q = \sum_{n=0}^{\infty} q^n/n!$. There is another expression $e^q = e^{q_a + \hat{\mathbf{v}}v} = e^{q_a}e^{\hat{\mathbf{v}}v} = e^{q_a}(\cos v + \hat{\mathbf{v}}\sin v)$ following from the properties of the exponential function $e^{q_1+q_2} = e^{q_1}e^{q_2}$, and the representation of $q = q_a + \hat{\mathbf{v}}v$ and $\hat{\mathbf{v}}^2 = -1$. In the last step, we have used

$$
\begin{aligned}
e^{\hat{\mathbf{v}}v} &= \sum_{n=0}^{\infty} \frac{(v\hat{\mathbf{v}})^n}{n!} \\
&= \sum_{m=0}^{\infty} \frac{(v\hat{\mathbf{v}})^{2m}}{(2m)!} + \sum_{m=0}^{\infty} \frac{(v\hat{\mathbf{v}})^{2m+1}}{(2m+1)!} \\
&= \sum_{m=0}^{\infty} \frac{(-1)^m v^{2m}}{(2m)!} + \hat{\mathbf{v}} \sum_{m=0}^{\infty} \frac{(-1)^m v^{2m+1}}{(2m+1)!} \\
&= \cos v + \hat{\mathbf{v}} \sin v
\end{aligned}
\tag{3.76}
$$

Making use of $e^q = e^{q_a}(\cos v + \hat{\mathbf{v}} \sin v)$, we have

$$
\frac{\partial e^q}{\partial q} = \frac{1}{2} \left(e^q + e^{q_a} v^{-1} \sin v \right). \tag{3.77}
$$

Example 2. The logarithmic function $f(q) = \ln q$ has representation

$$
\ln q = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n} (q-1)^n. \tag{3.78}
$$

with $a_n = (-1)^{n-1}/n$ and $q_0 = 1$. Since q_0 is a real number, $g(\tilde{q}^*) = f(q^*)$. Therefore, we have from Theorem 1

$$
\frac{\partial \ln q}{\partial q} = \frac{1}{2} \left(q^{-1} + \frac{\ln q - \ln q^*}{q - q^*} \right). \tag{3.79}
$$

Using representation $\ln q = \ln |q| + \hat{\mathbf{v}} \arccos(q_a/|q|)$, straightforwardly, the expression can be simplified as

$$
\frac{\partial \ln q}{\partial q} = \frac{1}{2} \left(q^{-1} + \frac{1}{v} \arccos \frac{q_a}{|q|} \right), \tag{3.80}
$$

where $v = |I(q)|$. If we derive the Eq. (3.79) directly, we can have

$$
\frac{\partial \ln q}{\partial q} = \frac{\partial \ln |q|}{\partial q} + \frac{\partial \hat{\mathbf{v}}}{\partial q} \arccos \frac{q_a}{|q|} + \hat{\mathbf{v}} \frac{\partial}{\partial q} (\arccos \frac{q_a}{|q|}) \tag{3.81}
$$

Moreover, the right side of above equation can be given as follows

$$\begin{aligned}
\frac{\partial \ln |q|}{\partial q} &= \frac{1}{|q|}\frac{\partial |q|}{\partial q} \\
&= \frac{1}{4|q|}(\frac{\partial |q|}{\partial q_a} - \frac{\partial |q|}{\partial q_b}i - \frac{\partial |q|}{\partial q_c}j - \frac{\partial |q|}{\partial q_d}k) \\
&= \frac{1}{8|q|}(\frac{\partial |q|^2}{\partial q_a} - \frac{\partial |q|^2}{\partial q_b}i - \frac{\partial |q|^2}{\partial q_c}j - \frac{\partial |q|^2}{\partial q_d}k) \\
&= \frac{q^*}{4|q|^2} \qquad (3.82)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial}{\partial q}(\arccos \frac{q_a}{|q|}) &= -\frac{1}{\sqrt{1-\frac{q_a^2}{|q|^2}}}\frac{\partial}{\partial q}\frac{q_a}{|q|} \\
&= -\frac{|q|}{\sqrt{|q|^2 - q_a^2}}(\frac{1}{|q|}\frac{\partial q_a}{\partial q} - \frac{q_a}{|q|^2}\frac{\partial |q|}{\partial q}) \\
&= -\frac{|q|}{\sqrt{|q|^2 - q_a^2}}(\frac{1}{4|q|} - \frac{q_a}{|q|^2}\frac{q^*}{4|q|}) \\
&= -\frac{1}{4}q^{-1}\hat{\mathbf{v}} \qquad (3.83)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial \hat{\mathbf{v}}}{\partial q} &= \frac{\partial}{\partial q}(\frac{I(q)}{|I(q)|}) \\
&= \frac{1}{|I(q)|}\frac{\partial I(q)}{\partial q} + I(q)(-\frac{1}{|I(q)|^2})\frac{\partial |I(q)|}{\partial q} \\
&= \frac{1}{|I(q)|}\frac{\partial I(q)}{\partial q} - \frac{1}{2|I(q)|^3})\frac{\partial |I(q)|^2}{\partial q} \\
&= \frac{1}{2|I(q)|} \qquad (3.84)
\end{aligned}$$

Adding these three terms together:

$$\begin{aligned}\frac{\partial \ln q}{\partial q} &= \frac{q^*}{4|q|^2} + \frac{1}{2|I(q)|}\arccos\frac{q_a}{|q|} + \hat{\mathbf{v}}(-\frac{1}{4}q^{-1}\hat{\mathbf{v}}) \\ &= \frac{1}{4}q^{-1} + \frac{1}{2|I(q)|}\arccos\frac{q_a}{|q|} + \frac{1}{4}q^{-1} \\ &= \frac{1}{2}q^{-1} + \frac{1}{2v}\arccos\frac{q_a}{|q|} \end{aligned} \tag{3.85}$$

Example 3. Hyperbolic tangent function $f(q) = \tanh q$ is defined as

$$\tanh q := \frac{e^q - e^{-q}}{e^q + e^{-q}} = q - \frac{q^3}{3} + \frac{2q^5}{15} - ... \tag{3.86}$$

Therefore, Theorem 1 applies. On the other hand, using the relation $e^q = e^{q_a}(\cos v + \hat{\mathbf{v}}\sin v)$, we can show that

$$\begin{aligned}\sinh(q) &= \frac{1}{2}(e^{q_a}(\cos v + \hat{\mathbf{v}}\sin v) - e^{-q_a}(\cos v - \hat{\mathbf{v}}\sin v)) \\ &= \sinh(q_a)\cos v + \hat{\mathbf{v}}\cosh(q_a)\sin v\end{aligned} \tag{3.87}$$

and

$$\begin{aligned}\cosh(q) &= \frac{1}{2}(e^{q_a}(\cos v + \hat{\mathbf{v}}\sin v) + e^{-q_a}(\cos v - \hat{\mathbf{v}}\sin v)) \\ &= \cosh(q_a)\cos v + \hat{\mathbf{v}}\sinh(q_a)\sin v.\end{aligned} \tag{3.88}$$

Then we can obtain

$$\tanh q = \frac{\sinh q}{\cosh q} = \frac{1}{2}\frac{\sinh 2q_a + \hat{\mathbf{v}}\sin 2v}{\sinh^2 q_a + \cos^2 v}. \tag{3.89}$$

Then derivatives in the expression given by Theorem 1 can be simplified. The final expression can be written as

$$\frac{\partial \tanh q}{\partial q} = \frac{1}{2}\left(\operatorname{sech}^2 q + \frac{v^{-1}\sin 2v}{\cosh 2q_a + \cos 2v}\right), \tag{3.90}$$

where $\operatorname{sech} q := 1/\cosh q$ is the quaternionic hyperbolic secant function.

Remark. Apparently, the derivatives for these functions can also be found by direct calculations without resorting to Theorem 1.

We now turn to a question of more theoretical interests. Even though it might not be obvious from the definitions, the following theorem shows that the restricted HR derivative is consistent with the derivative in the real domain for a class of functions, including those in the above examples.

Theorem 2. For the function $f(q)$ in Theorem 1, if q_0 is a real number, then

$$\frac{\partial f(q)}{\partial q} \to f'(q) \tag{3.91}$$

when $q \to R(q)$, i.e., when q approaches a real number.

Proof. Using the polar representation, we write $\tilde{q} = |\tilde{q}| \exp(\hat{\mathbf{v}}\theta)$, where $\theta = \arcsin(v/|\tilde{q}|)$ is the argument of $\tilde{q}$ with $v = |I(\tilde{q})|$. Then $\tilde{q}^n = |\tilde{q}|^n \exp(n\hat{\mathbf{v}}\theta)$, and

$$(\tilde{q}^n - \tilde{q}^{*n})(\tilde{q} - \tilde{q}^*)^{-1} = \frac{I(\tilde{q}^n)}{I(\tilde{q})} = \frac{|\tilde{q}|^{n-1}\sin(n\theta)}{\sin\theta}. \tag{3.92}$$

For real q_0, $\tilde{q} \to q_a - q_0$ and $v \to 0$ when $q \to R(q)$. There are two possibilities. Firstly, if $q_a - q_0 \geq 0$, then $\theta \to 0$ at the limit. Thus,

$$\begin{gathered} \frac{\sin(n\theta)}{\sin\theta} \sim \frac{\sin(n\theta)}{\theta} \to n, \\ |\tilde{q}|^{n-1} \to (q_a - q_0)^{n-1}. \end{gathered} \tag{3.93}$$

Therefore,

$$(\tilde{q}^n - \tilde{q}^{*n})(\tilde{q} - \tilde{q}^*)^{-1} \to n\tilde{q}^{n-1} \tag{3.94}$$

$$[g(\tilde{q}) - g(\tilde{q}^*)](\tilde{q} - \tilde{q}^*)^{-1} \to \sum_{n=-\infty}^{\infty} n a_n \tilde{q}^{n-1} = f'(q). \tag{3.95}$$

Thus,

$$\frac{\partial f(q)}{\partial q} \to \frac{1}{2}[f'(q) + f'(q)] = f'(q). \tag{3.96}$$

Secondly, if $q_a - q_0 < 0$, then $\theta \to \pi$. Thus

$$\frac{\sin(n\theta)}{\sin\theta} \sim \frac{\sin(n\theta)}{\pi - \theta} \tag{3.97}$$

Note $\sin(n\theta) = \sin[n\pi - n(\pi - \theta)] = (-1)^{n-1}\sin[n(\pi - \theta)]$, we have

$$\frac{\sin(n\theta)}{\sin\theta} \sim \frac{(-1)^{n-1}\sin(n(\pi - \theta)}{\pi - \theta} \to (-1)^{n-1}n. \tag{3.98}$$

On the other hand, in this case $|\tilde{q}| \to -(q_a - q_0)$, hence

$$|\tilde{q}|^{n-1} \to (-1)^{n-1}(q_a - q_0)^{n-1}. \tag{3.99}$$

Since $\tilde{q} \to q_a - q_0$, as a consequence, we have

$$(\tilde{q}^n - \tilde{q}^{*n})(\tilde{q} - \tilde{q}^*)^{-1} \to n\tilde{q}^{n-1} \tag{3.100}$$

which is the same as Eq. (3.94). The proof then follows from the first case.

The functions in above three examples all satisfy the conditions in Theorem 2, hence we expect Theorem 2 applies. One can easily verify by direct calculations that the theorem indeed holds.

3.5 The Right Restricted HR Gradients

In this section, we briefly summarize the results for the right restricted HR gradients, and highlight the difference with left restricted HR gradients.

1. Right-linearity: for arbitrary quaternion constants α and β, and functions $f(q)$ and $g(q)$, we have

$$\frac{\partial^R(f\alpha + g\beta)}{\partial q^\nu} = \frac{\partial^R f}{\partial q^\nu}\alpha + \frac{\partial^R g}{\partial q^\nu}\beta. \tag{3.101}$$

However, linearity does not hold for left multiplications, i.e., in general $\frac{\partial^R \alpha f}{\partial q} \neq \alpha \frac{\partial^R f}{\partial q}$.

2. The first product rule: for the right restricted HR operator, the following product rule holds

$$[\nabla_q^R(fg)]^T = [(\nabla_q^R f)g]^T + J^*[f(\nabla_r g)^T] \,. \tag{3.102}$$

The second and third product rules are the same as for the left restricted operator.

3. The first chain rule: for the composite function $f(g(q))$, we have

$$(\nabla_q^R f)^T = M^T(\nabla_q^{gR} f)^T. \tag{3.103}$$

4. The second chain rule becomes $(\nabla_q^R f)^T = O^T(\nabla_r^g f)^T$.

5. The third chain rule becomes $\frac{\partial^R f}{\partial q^\nu} = \frac{\partial g}{\partial q^\nu}\frac{\partial f}{\partial g}$. Note that, $\partial g/\partial q^\nu = \partial^R g/\partial q^\nu$ since g is real-valued. We thus have omitted the superscript R. Also, $\partial f/\partial g$ is a real derivative, so there is no distinction between left and right derivatives.

We can also find the right restricted HR gradients for common quaternion functions. First of all, Lemma 1 is also true for right derivatives:

Lemma 2. For $f(q) = (q - q_0)^n$ with n integer and q_0 a constant quaternion, we have

$$\frac{\partial^R f(q)}{\partial q} = \frac{1}{2}\left(n\tilde{q}^{n-1} + \frac{\tilde{q}^n - \tilde{q}^{*n}}{\tilde{q} - \tilde{q}^*}\right), \tag{3.104}$$

with $\tilde{q} = q - q_0$.

Remark. To prove the lemma, we use the following recurrent relations:

$$\frac{\partial (q-q_0)^n}{\partial q} = \frac{\partial \tilde{q}^{n-1}}{\partial q}\tilde{q} + R(\tilde{q}^{n-1}) \tag{3.105}$$

$$\frac{\partial ((q-q_0)^{-n})}{\partial q} = \left[\frac{\partial \tilde{q}^{-(n-1)}}{\partial q} - R(\tilde{q}^{-n})\right]\tilde{q}^{-1}. \tag{3.106}$$

Using Lemma 2, We can prove the following result:

Theorem 3. Assuming $f : H \to H$ admits a power series representation $f(q) := g(\tilde{q}) := \sum_{n=-\infty}^{\infty} \tilde{q}^n a_n$, with a_n being a quaternion constant and $\tilde{q} = q - q_0$, for $R_1 \leq |\tilde{q}| \leq R_2$ with $R_1, R_2 > 0$ being some constants, then

$$\frac{\partial^R f(q)}{\partial q} = \frac{1}{2}\left[f'(q) + (\tilde{q} - \tilde{q}^*)^{-1}(g(\tilde{q}) - g(\tilde{q}^*))\right], \tag{3.107}$$

where $f'(q)$ is the derivative in the usual sense, i.e.,

$$f'(q) := \sum_{n=-\infty}^{\infty} n\tilde{q}^{n-1}a_n = \sum_{n=-\infty}^{\infty} n(q-q_0)^{n-1}a_n. \tag{3.108}$$

Note that, the functions $f(q)$ in Theorem 3 in general form a different class of functions than the one in Theorem 1, because in the series representation a_n appears on the right-hand side of the powers. However, if a_n is a real number, then the two classes of functions coincide. Therefore, we have the following result:

Theorem 4. If a_n is real, then the left and right restricted HR gradients of $f(q)$ coincide.

Remark. As a consequence, we can see immediately the right derivatives for the exponential, logarithmic and hyperbolic tangent functions are the same as the left ones.

Apparently, Theorem 2 is also true for the right derivatives. Hence, we have:

Theorem 5. The right-restricted HR gradient is consistent with the real gradient in the sense of Theorem 2.

3.6 Quaternion-valued Algorithms

3.6.1 The increment of a quaternion function

When $f(q)$ is a real-valued quaternion function, both left and right restricted HR gradients are coincident with the HR gradients. Besides, we have

$$\frac{\partial^R f}{\partial q^\nu} = \frac{\partial f}{\partial q^\nu} = \left(\frac{\partial f}{\partial q}\right)^\nu, \tag{3.109}$$

where $\nu \in \{1, i, j, k\}$. Thus only $\partial f / \partial q$ is independent. As a consequence (see also [42]),

$$\begin{aligned} df &= \sum_\nu \frac{\partial f}{\partial q^\nu} dq^\nu = \sum_\nu \left(\frac{\partial f}{\partial q}\right)^\nu dq^\nu \\ &= \sum_\nu \left(\frac{\partial f}{\partial q} dq\right)^\nu = 4R\left(\frac{\partial f}{\partial q} dq\right), \end{aligned} \tag{3.110}$$

where equation (3.109) has been used. Hence, $-(\partial f / \partial q)^*$ gives the steepest descent direction for f, and the increment is determined by $\partial f / \partial q$.

On the other hand, if f is a quaternion-valued function, the increment will depend on all four derivatives. Taking $f(q) = q^2$ as an example, we have (see equations (3.57) and (3.59))

$$dq^2 = (q + q_a)dq + q_b i dq^i + q_c j dq^j + q_d k dq^k, \tag{3.111}$$

even though $f(q)$ appears to be independent of q^i, q^j and q^k. It can be verified that the above expression is the same as the differential form given in terms of dq_a, dq_b, dq_c and dq_d. Thus it is essential to include the contributions from $\partial f/\partial q^i$ etc.

We also note that, if the right gradient is used consistently, the same increment would result, since the basis of the definitions is the same, namely, the differential form in term of dq_a, dq_b, dq_c and dq_d.

We have proposed a restricted HR gradient operator and discussed its properties, in particular several different versions of product rules and chain rules. Using the rules that we establish, we can derive a general formula for the derivative of a large class of nonlinear quaternion-valued functions. The class includes the common elementary functions such as the exponential function, the logarithmic function, among others. We also prove that, for a wide class of functions, the restricted HR gradient becomes the usual derivatives for real functions with respect to real variables, when the independent quaternion variable tends to the real axis, thus showing the consistency of the definition. Both linear and nonlinear adaptive filtering algorithms could be derived to show the applications of the gradient operator later.

3.6.2 The QLMS algorithm

As the gradient operator expressions for quaternion variable q and its conjugate q^* were given before

$$\frac{\partial f(q)}{\partial q} = \frac{1}{4}(\frac{\partial f(q)}{\partial q_a} - \frac{\partial f(q)}{\partial q_b}i - \frac{\partial f(q)}{\partial q_c}j - \frac{\partial f(q)}{\partial q_d}k) \tag{3.112}$$

$$\frac{\partial f(q)}{\partial q^*} = \frac{1}{4}(\frac{\partial f(q)}{\partial q_a} + \frac{\partial f(q)}{\partial q_b}i + \frac{\partial f(q)}{\partial q_c}j + \frac{\partial f(q)}{\partial q_d}k), \tag{3.113}$$

when the quaternion variable q is replaced by a quaternion-valued vector $\mathbf{w} = [w_1\ w_2\ \cdots\ w_M]^T$, where $w_m = a_m + b_m i + c_m j + d_m k$, $m = 1, ..., M$, the differentiation of $f(\mathbf{w})$ with respect to $\mathbf{w}$ can be derived as follows

$$\frac{\partial f}{\partial \mathbf{w}} = \frac{1}{4}\begin{bmatrix} \frac{\partial f}{\partial a_1} - \frac{\partial f}{\partial b_1}i - \frac{\partial f}{\partial c_1}j - \frac{\partial f}{\partial d_1}k \\ \frac{\partial f}{\partial a_2} - \frac{\partial f}{\partial b_2}i - \frac{\partial f}{\partial c_2}j - \frac{\partial f}{\partial d_2}k \\ \vdots \\ \frac{\partial f}{\partial a_M} - \frac{\partial f}{\partial b_M}i - \frac{\partial f}{\partial c_M}j - \frac{\partial f}{\partial d_M}k \end{bmatrix} \tag{3.114}$$

Similarly, we define $\frac{\partial f}{\partial \mathbf{w}^*}$ as

$$\frac{\partial f}{\partial \mathbf{w}^*} = \frac{1}{4}\begin{bmatrix} \frac{\partial f}{\partial a_1} + \frac{\partial f}{\partial b_1}i + \frac{\partial f}{\partial c_1}j + \frac{\partial f}{\partial d_1}k \\ \frac{\partial f}{\partial a_2} + \frac{\partial f}{\partial b_2}i + \frac{\partial f}{\partial c_2}j + \frac{\partial f}{\partial d_2}k \\ \vdots \\ \frac{\partial f}{\partial a_M} + \frac{\partial f}{\partial b_M}i + \frac{\partial f}{\partial c_M}j + \frac{\partial f}{\partial d_M}k \end{bmatrix} \tag{3.115}$$

As an application, we now apply the quaternion-valued restricted HR gradient operator to develop the QLMS algorithm. Different versions of the QLMS algorithm have been derived before. However, with the rules we have derived, some of the calculations can be simplified, as we will be showing below.

In terms of a standard adaptive filter, the output $y[n]$ and error $e[n]$ can be expressed as

$$y[n] = \mathbf{w}^T[n]\mathbf{x}[n], \;\; e[n] = d[n] - \mathbf{w}^T[n]\mathbf{x}[n], \tag{3.116}$$

where $\mathbf{w}[n]$ is the adaptive weight coefficient vector, $d[n]$ the reference signal, and $\mathbf{x}[n]$ the input sample vector. The conjugate $\mathbf{e}^*[n]$ of the error signal $e[n]$ is

$$e^*[n] = d^*[n] - \mathbf{x}^H[n]\mathbf{w}^*[n]. \tag{3.117}$$

The cost function is defined as $J_2[n] = e[n]e^*[n]$ which is real-valued. According to the discussion above and in [42, 54], the conjugate gradient $(\nabla_{\mathbf{w}} J_2[n])^*$ gives the maximum steepness direction for the optimization surface. Therefore it is used to update the weight vector. Specifically,

$$\mathbf{w}[n+1] = \mathbf{w}[n] - \mu_2(\nabla_{\mathbf{w}} J_2[n])^*, \tag{3.118}$$

where μ_3 is the step size. When expanding the cost function, we obtain

$$\begin{aligned} J_2[n] &= e[n]e^*[n] \\ &= d[n]d^*[n] - d[n]\mathbf{x}^H[n]\mathbf{w}^*[n] - \mathbf{w}^T[n]\mathbf{x}[n]d^*[n] + \mathbf{w}^T[n]\mathbf{x}[n]\mathbf{x}^H[n]\mathbf{w}^*[n]. \end{aligned} \tag{3.119}$$

Details of the derivation process for the gradient are shown in the following without using our rules mentioned before

$$\frac{\partial(d[n]d^*[n])}{\partial \mathbf{w}^*[n]} = 0 \tag{3.120}$$

$$\frac{\partial(d[n]\mathbf{x}^H[n]\mathbf{w}^*[n])}{\partial \mathbf{w}^*[n]} = d[n]\mathbf{x}^*[n] \tag{3.121}$$

$$\frac{\partial(\mathbf{w}^T[n]\mathbf{x}[n]d^*[n])}{\partial\mathbf{w}^*[n]} = -\frac{1}{2}d[n]\mathbf{x}^*[n] \tag{3.122}$$

$$\frac{\partial(\mathbf{w}^T[n]\mathbf{x}[n]\mathbf{x}^H[n]\mathbf{w}^*[n])}{\partial\mathbf{w}^*[n]} = \frac{1}{2}\mathbf{w}^T[n]\mathbf{x}[n]\mathbf{x}^*[n]. \tag{3.123}$$

Combining the above results, the final gradient can be obtained as follows

$$\nabla_{\mathbf{w}^*}J_2[n] = -\frac{1}{2}e[n]\mathbf{x}^*[n]. \tag{3.124}$$

To find $\nabla_{\mathbf{w}}J_2[n]$, we can also use another simpler way with employing the first product rule:

$$\begin{aligned}\nabla_{\mathbf{w}}J_2[n] &= \frac{\partial e[n]e^*[n]}{\partial\mathbf{w}} \\ &= e[n]\frac{\partial e^*[n]}{\partial\mathbf{w}} + \frac{1}{4}(\frac{\partial e[n]}{\partial\mathbf{w}_a}e^*[n] - \frac{\partial e[n]}{\partial\mathbf{w}_b}e^*[n]i \\ &\quad -\frac{\partial e[n]}{\partial\mathbf{w}_c}e^*[n]j - \frac{\partial e[n]}{\partial\mathbf{w}_d}e^*[n]k)\end{aligned} \tag{3.125}$$

After some algebra, we find

$$\nabla_{\mathbf{w}}J_2[n] = -\frac{1}{2}\mathbf{x}[n]e^*[n], \tag{3.126}$$

which is the same result as the extended way, thereby leading to the following update equation for the QLMS algorithm

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu_2(e[n]\mathbf{x}^*[n]). \tag{3.127}$$

3.6.3 The AQLMS algorithm

Recently, to fully exploit the second-order statistics of the signals, an augmented formulation of the data vector has been proposed, first for complex-valued signals and then for quaternion-valued ones. For complex-valued signals, the augmented vector is composed of the original data and its conjugate, while for the latter, due to existence of the three quaternion involutions, the choice for the augmented vector is not unique. Without loss of generality, here we adopt the simplest formulation by combining the data vector $\mathbf{x}[n]$ and its conjugate $\mathbf{x}^*[n]$ to produce an augmented vector $\mathbf{x}_a[n] = \left[\mathbf{x}^T[n] \;\; \mathbf{x}^H[n]\right]^T$ [55], where $\{\cdot\}^H$ is a combination of the operations of $\{\cdot\}^T$ and $\{\cdot\}^*$ for a quaternion. For such a "widely linear" model, the quaternion-valued output for the conjugate part of the input is given by

$$\hat{y}[n] = \mathbf{g}^T[n]\mathbf{x}^*[n], \tag{3.128}$$

where $\mathbf{g}[n]$ denotes the weight vector for the conjugate part of the input $\mathbf{x}[n]$.

As to the AQLMS algorithm, the update of the weight vector of the conjugate part $\mathbf{g}[n]$ can be found with the same method as that of the QLMS, i.e.

$$\mathbf{g}[n+1] = \mathbf{g}[n] + \mu(e[n]\mathbf{x}[n]). \tag{3.129}$$

With the augmented weight vector $\mathbf{h}_a[n]$ defined as

$$\mathbf{h}_a[n] = \left[\mathbf{w}^T[n] \;\; \mathbf{g}^T[n]\right]^T, \tag{3.130}$$

we obtain the following update equation

$$\mathbf{h}_a[n+1] = \mathbf{h}_a[n] + \mu_3(e_a[n]\mathbf{x}_a{}^*[n]) \tag{3.131}$$

where $e_a[n] = d[n] - \mathbf{h}_a{}^T[n]\mathbf{x}_a[n]$.

3.6.4 The Quaternion-valued nonlinear adaptive algorithm

Another application of the proposed gradient operator is the derivation of non-linear quaternion-valued adaptive filtering algorithms. We use the quaternion-valued hyperbolic tangent function as an example [56], so that the output $s[n]$ of the adaptive filter is given by

$$s[n] = \tanh(y[n]) = \tanh(\mathbf{w}^T[n]\mathbf{x}[n]). \tag{3.132}$$

The cost function is given by

$$J_4[n] = e[n]e^*[n], \tag{3.133}$$

with

$$e[n] = d[n] - \tanh(\mathbf{w}^T[n]\mathbf{x}[n]). \tag{3.134}$$

Using the product rules in (3.125) and chain rules, and letting $y[n] = \mathbf{w}^T[n]\mathbf{x}[n]$, we have

$$\begin{aligned}\frac{\partial e^*[n]}{\partial \mathbf{w}[n]} = &-(\frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_a}\frac{\partial (y^*[n])_a}{\partial \mathbf{w}[n]} + \frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_b}\frac{\partial (y^*[n])_b}{\partial \mathbf{w}[n]} \\ &+ \frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_c}\frac{\partial (y^*[n])_c}{\partial \mathbf{w}[n]} + \frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_d}\frac{\partial (y^*[n])_d}{\partial \mathbf{w}[n]}).\end{aligned} \tag{3.135}$$

Let $u = |I(y)|$ and $\hat{\mathbf{u}} = I(y)/u$. Then the quaternion $y = y_a + I(y)$ can also be written as $y = y_a + u\hat{\mathbf{u}}$. $\hat{\mathbf{u}}$ is a pure unit quaternion. If we employ the following expression for $\tanh y$

$$\tanh y = \frac{1}{2}\frac{\sinh 2y_a + \hat{\mathbf{u}}\sin 2u}{\sinh^2 y_a + \cos^2 u}, \tag{3.136}$$

finally, the gradient can be expressed as follows by using (3.89)

$$\begin{aligned}\nabla_{\mathbf{w}} J_4[n] = \frac{1}{4(\sinh^2 y_a + \cos^2 u)^2}\big((2\sin 2u(e_a \sin^2 y_a + \sin 2u(e\hat{\mathbf{u}})_a)\\ + (\cos u - \frac{\sin u}{u})(\sinh^2 y_a + \cos^2 u)(e\hat{\mathbf{u}})_a)\mathbf{x}\hat{\mathbf{u}}\\ + e_a((\sinh^2 y_a + \cos^2 u)(\frac{\sin u}{u} - 4\cosh 2y_a)\\ + \sinh 2y_a(\sinh^2 y_a - \sin 2u(e\hat{\mathbf{u}})_a))\mathbf{x}\\ + 2\frac{\sin u}{u}(\sinh^2 y_a + \cos^2 u)(e\mathbf{x}_a + e^*\mathbf{x})_a)\big) \end{aligned} \tag{3.137}$$

Substituting the above result into Eq. (3.118) we can then obtain the update equation for the nonlinear adaptive algorithm.

On the other hand, if we use the series representation of $\tanh(q)$, we can obtain another form of the gradient function. Using the product rules, the gradient of the cost function is given by:

$$\begin{aligned}\nabla_{\mathbf{w}} J_4[n] = \frac{\partial J_4[n]}{\partial \mathbf{w}[n]} = e[n]\frac{\partial e^*[n]}{\partial \mathbf{w}[n]}\\ + \frac{1}{4}\left(\frac{\partial e[n]}{\partial w_a}e^*[n] - \frac{\partial e[n]}{\partial w_b}e^*[n]i - \frac{\partial e[n]}{\partial w_c}e^*[n]j - \frac{\partial e[n]}{\partial w_d}e^*[n]k\right)\end{aligned} \tag{3.138}$$

Letting $y[n] = \mathbf{w}^T[n]\mathbf{x}[n]$, we have

$$\begin{aligned}\frac{\partial e^*[n]}{\partial \mathbf{w}[n]} &= -\frac{\partial \tanh(y^*[n])}{\partial \mathbf{w}[n]}\\ &= -(\frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_a}\frac{\partial (y^*[n])_a}{\partial \mathbf{w}[n]} + \frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_b}\frac{\partial (y^*[n])_b}{\partial \mathbf{w}[n]}\\ &\quad + \frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_c}\frac{\partial (y^*[n])_c}{\partial \mathbf{w}[n]} + \frac{\partial \tanh(y^*[n])}{\partial (y^*[n])_d}\frac{\partial (y^*[n])_d}{\partial \mathbf{w}[n]})\\ &= \frac{1}{2}\sum_{m=0}^{\infty}\sum_{r=0}^{m-1} a_m (y^*[n])^{m-1-r}\mathbf{x}^*[n](y[n])^r, \end{aligned} \tag{3.139}$$

and

$$\begin{aligned}&\frac{1}{4}\left(\frac{\partial e[n]}{\partial w_a}e^*[n]-\frac{\partial e[n]}{\partial w_b}e^*[n]i-\frac{\partial e[n]}{\partial w_c}e^*[n]j-\frac{\partial e[n]}{\partial w_d}e^*[n]k\right)\\&=-\sum_{m=0}^{\infty}\sum_{r=0}^{m-1}a_m(y[n])^{m-1-r}R(\mathbf{x}[n](y[n])^r e^*[n]),\end{aligned} \tag{3.140}$$

where a_m is the coefficient in the series representation of $\tanh(y[n])$,

$$\tanh(y[n])=\sum_{m=0}^{\infty}a_m(y[n])^m. \tag{3.141}$$

The gradient can then be expressed as

$$\begin{aligned}\nabla_{\mathbf{w}}J[n] &= \frac{1}{2}\sum_{m=0}^{\infty}\sum_{r=0}^{m-1}a_m e[n](y^*[n])^{m-1-r}\mathbf{x}^*[n](y[n])^r\\&\quad-\sum_{m=0}^{\infty}\sum_{r=0}^{m-1}a_m(y[n])^{m-1-r}R(\mathbf{x}[n](y[n])^r e^*[n])\\&= -\frac{1}{2}\sum_{m=0}^{\infty}\sum_{r=0}^{m-1}a_m(\mathbf{x}[n](\mathbf{w}^T[n]\mathbf{x}[n])^r e^*[n](\mathbf{w}^T[n]\mathbf{x}[n])^{m-1-r}\end{aligned} \tag{3.142}$$

As the conjugate gradient gives the maximum steepness direction for the optimization surface, the hyperbolic tangent function based nonlinear adaptive filtering algorithm is given by:

$$\begin{aligned}\mathbf{w}[n+1] &= \mathbf{w}[n]+\frac{1}{2}\mu\sum_{m=0}^{\infty}a_m\big(\sum_{r=0}^{m-1}(\mathbf{x}^H[n]\mathbf{w}^*[n])^{m-1-r}e[n](\mathbf{x}^H[n]\mathbf{w}^*[n])^r\big)\mathbf{x}^*[n]\\&= \mathbf{w}[n]+\frac{1}{2}\mu\big(\sum_{m=0}^{\infty}a_m h_m\big)\mathbf{x}^*[n],\end{aligned} \tag{3.143}$$

where

$$h_m=\sum_{r=0}^{m-1}(\mathbf{x}^H[n]\mathbf{w}^*[n])^{m-1-r}e[n](\mathbf{x}^H[n]\mathbf{w}^*[n])^r. \tag{3.144}$$

From the above equation, we can see that if we allow $e[n]$ to be commuted with the other factors, then the summation becomes $sech^2(\mathbf{x}^H[n]\mathbf{w}^*[n])$, and the formula is the same as in the real or complex domains.

3.7 A Zero-attracting QLMS Algorithm for Sparse System Identification

In adaptive filtering [21], there is a class of algorithms specifically designed for sparse system identification, where the unknown system only has a few large coefficients while the remaining ones have a very small amplitude so that they can be ignored without significant effect on the overall performance of the system. A good example of them is the zero-attracting least mean square (ZA-LMS) algorithm proposed in [19]. This algorithm can achieve a higher convergence speed, and meanwhile, reduce the steady state excess mean square error (MSE). Compared to the classic LMS algorithm [24], the ZA-LMS algorithm introduces a l_1 norm in its cost function, which modifies the update equation of weight coefficient vector with a zero attractor term.

In this part, we propose a novel quaternion-valued adaptive algorithm with a sparsity constraint, which is called zero-attracting QLMS (ZA-QLMS) algorithm. The additional constraint is formulated based on the l_1 norm. Both the QLMS and ZA-QLMS algorithms can identify an unknown sparse system effectively. However, a better performance in terms of convergence speed is achieved by the latter one.

3.7.1 The zero-attracting QLMS (ZA-QLMS) algorithm

To derive the ZA-QLMS algorithm, similar to [19], in the cost function, we add a l_1 norm penalty term for the quaternion-valued weight vector $\mathbf{w}[n]$.

For a standard adaptive filter, the output $y[n]$ and error $e[n]$ can be expressed as

$$
\begin{aligned}
y[n] &= \mathbf{w}^T[n]\mathbf{x}[n] && (3.145)\\
e[n] &= d[n] - \mathbf{w}^T[n]\mathbf{x}[n]\ . && (3.146)
\end{aligned}
$$

The conjugate form $\mathbf{e}^*[n]$ of the error signal $e[n]$ is given by

$$e^*[n] = d^*[n] - \mathbf{x}^H[n]\mathbf{w}^*[n]. \tag{3.147}$$

Our proposed cost function with a zero attractor term is then formulated as

$$J_5[n] = e[n]e^*[n] + \gamma_1\|\mathbf{w}[n]\|_1\ , \tag{3.148}$$

where γ_1 is a small constant.

The gradient of the above cost function with respect to $\mathbf{w}^*[n]$ and $\mathbf{w}[n]$ can be respectively expressed as

$$\nabla_{\mathbf{w}^*} J_5[n] = \frac{\partial J_5[n]}{\partial \mathbf{w}^*} \tag{3.149}$$

and

$$\nabla_{\mathbf{w}} J_5[n] = \frac{\partial J_5[n]}{\partial \mathbf{w}} \tag{3.150}$$

From [42, 54], we know that the conjugate gradient gives the maximum steepness direction for the optimization surface. Therefore, the conjugate gradient $\nabla_{\mathbf{w}^*} J_5[n]$ will be used to derive the update of the coefficient weight vector.

By using our proposed product rules, the derivation process for the gradient is simplified and expressed as

$$
\begin{aligned}
\frac{\partial J_5[n]}{\partial \mathbf{w}^*} &= \frac{\partial(e[n]e^*[n] + \gamma_1\|\mathbf{w}[n]\|_1)}{\partial \mathbf{w}^*}\\
&= \frac{\partial(e[n]e^*[n])}{\partial \mathbf{w}^*} + \frac{\partial(\gamma_1\|\mathbf{w}[n]\|_1)}{\partial \mathbf{w}^*}\ . && (3.151)
\end{aligned}
$$

The first part of the above equation is same as that for QLMS algorithm, which is

$$\nabla_{\mathbf{w}^*} J_2[n] = -\frac{1}{2} e[n]\mathbf{x}^*[n]. \tag{3.152}$$

Moreover, the last part of the gradient of cost function is given by

$$\frac{\partial(\gamma_1 \|\mathbf{w}[n]\|_1)}{\partial \mathbf{w}^*} = \frac{1}{4}\gamma_1 \cdot sgn(\mathbf{w}[n]) \ , \tag{3.153}$$

where the symbol sgn is a component-wise sign function that is defined as [19]

$$sgn(x) = \begin{cases} x/|x| & x \neq 0 \\ 0 & x = 0 \end{cases}$$

Combining the above results, the final gradient can be obtained as follows

$$\nabla_{\mathbf{w}^*} J_5[n] = -\frac{1}{2} e[n]\mathbf{x}^*[n] + \frac{1}{4}\gamma_1 \cdot sgn(\mathbf{w}[n]) \ . \tag{3.154}$$

With the general update equation for the weight vector

$$\mathbf{w}[n+1] = \mathbf{w}[n] - \mu_5 \nabla_{\mathbf{w}^*} J_5[n], \tag{3.155}$$

where μ_5 is the step size, we arrive at the following update equation for the proposed ZA-QLMS algorithm

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \frac{1}{2}\mu_5 (e[n]\mathbf{x}^*[n]) - \frac{1}{4}\rho_1 \cdot sgn(\mathbf{w}[n]) \ , \tag{3.156}$$

where $\rho_1 = \mu_5\gamma_1$. The last term represents the zero attractor, which enforces the near-zero coefficients to zero and therefore accelerates the convergence process when majority of the system coefficients are nearly zero in a sparse system.

Note that equation (3.156) will be reduced to the normal QLMS algorithm without the zero attractor term, given by

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu_2(e[n]\mathbf{x}^*[n]) \ , \tag{3.157}$$

and the cost function is changed to

$$J_2[n] = e[n]e^*[n] \ . \tag{3.158}$$

3.7.2 Simulation results

In this part, simulations are performed for sparse system identification using the proposed algorithm in comparison with the QLMS algorithm. The sparse systems are considered in the following. The input signal to the adaptive filter is colored and generated by passing a quaternion-valued white gaussian signal through a randomly generated filter. The noise part is quaternion-valued white Gaussian and added to the output of the unknown sparse system, with a 30dB signal to noise ratio (SNR) for both scenarios.

Scenario One

For this scenario, the parameters are: the step size is 4×10^{-7}; the unknown sparse FIR filter length L is 32, with 4 non-zero coefficients at the 2nd, 8th, 16th and 31st taps, and its magnitude of the impulse response is shown in Fig. 3.1; the coefficient of the zero attractor ρ_1 is 1×10^{-6}. The learning curve obtained by averaging 100 runs of the corresponding algorithm is given in Fig. 3.2, where we can see that the ZA-QLMS algorithm has achieved a faster convergence speed than the QLMS algorithm when they both reach a similar steady state.

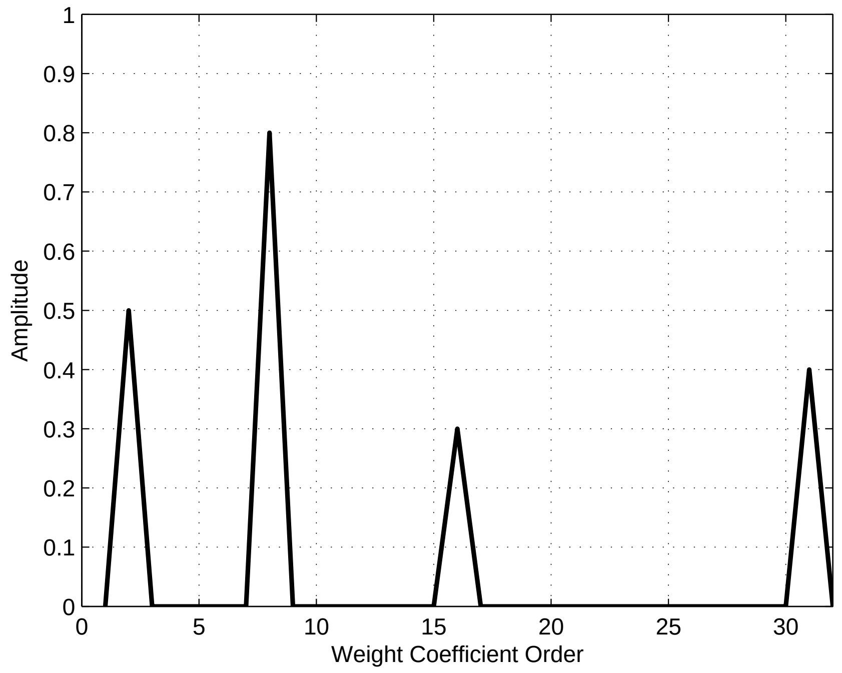

Fig. 3.1: Magnitude of the impulse response of the sparse system.

Scenario Two

For this case, length of the unknown FIR filter is reduced to 16, still with 4 active taps. The parameters are: step size is 6×10^{-7} and the value of ρ_1 is 1×10^{-6}. The results are shown in Fig. 3.3. Again we see that the ZA-QLMS algorithm has a faster convergence speed and has even converged to a lower steady state error in this specific scenario. With zero attractor term still outperforms the QLMS algorithm in faster convergence speed for the sparse system identification, even with the different sparsity from the first scenario. Furthermore, the ZA-QLMS

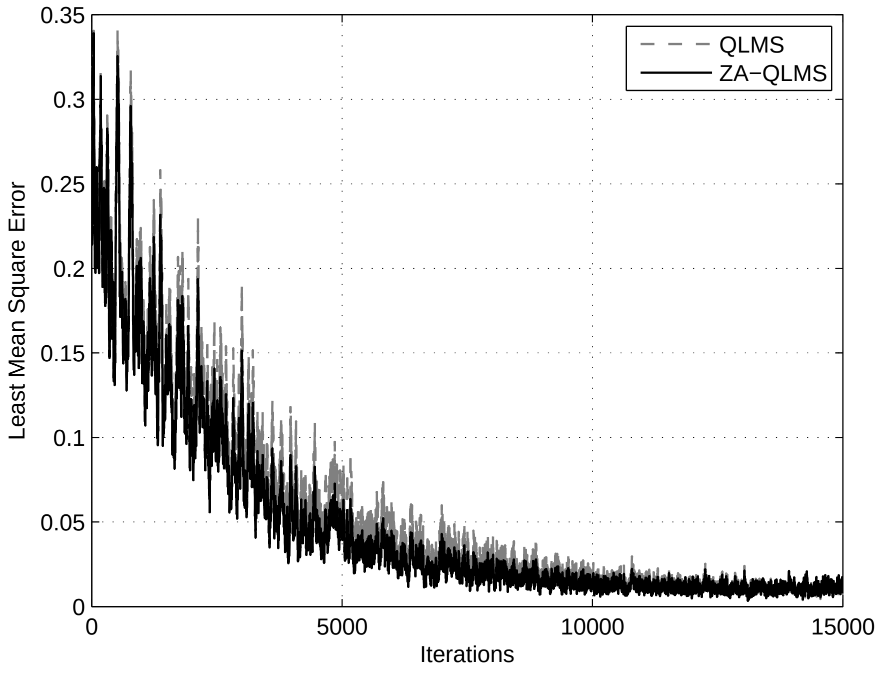

Fig. 3.2: Learning curves for the first scenario.

algorithm gets smaller mean square error at the later stage.

3.8 Summary

In this chapter, we have given a detailed derivation of two gradient operators: the restricted HR gradient operator and the right restricted HR gradient operator. We also documented some properties of the operators, in particular several different versions of product rules and chain rules. After that, we prove that the restricted HR gradients are consistent with the usual definition for the gradient of a real function of a real variable.

Then, application of the gradient operator to the derivation of QLMS adap-

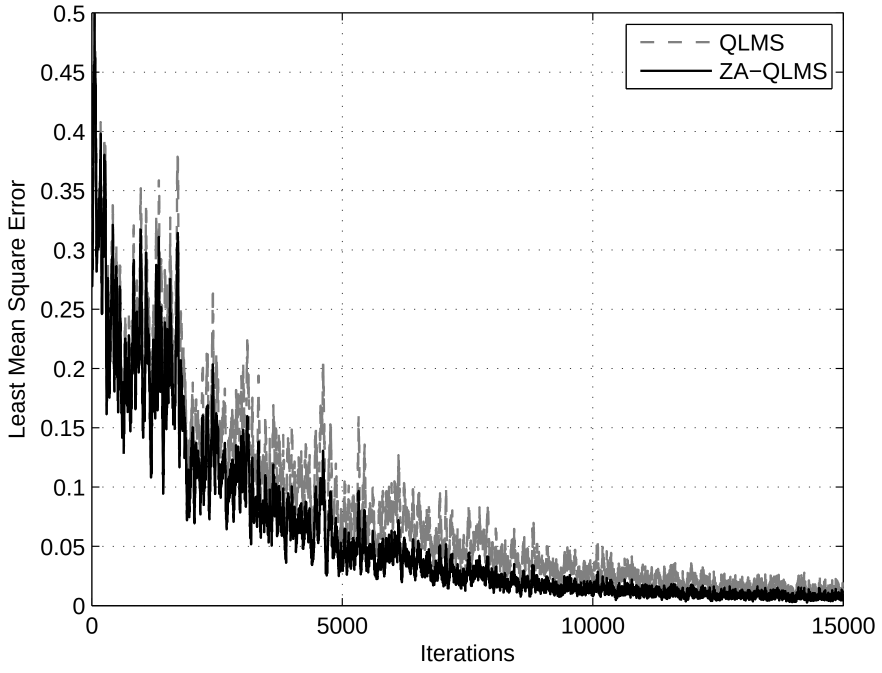

Fig. 3.3: Learning Curves for the second scenario.

tive algorithm and a nonlinear adaptive algorithm based on the hyperbolic tangent function is discussed. Furthermore, a quaternion-valued adaptive algorithm has been proposed for more efficient identification of unknown sparse systems. It is derived by introducing an l_1 penalty term in the original cost function and the resultant ZA-QLMS algorithm can achieve a faster convergence rate by incorporating the sparsity information of the system into the update process. Simulation results have been provided to show the effectiveness of the new algorithm with a zero attractor.

Chapter 4

Application to Adaptive Beamforming

Adaptive beamforming has a range of applications and has been studied extensively in the past in various areas ranging from radar, sonar, microphone arrays, radio astronomy, seismology, medical diagnosis and treatment, to communications [22, 57, 58, 59, 60, 61, 62, 63, 64, 65], especially for traditional sensor arrays. With the introduction of vector sensor arrays, such as those consisting of crossed-dipoles and tripoles [66, 67, 68], adaptive beamforming for such an array system has attracted more and more attention recently [2, 11, 67].

In this work, we consider the crossed-dipole array and study the problem of how to reduce the number of sensors involved in the adaptive beamforming process so that reduced system complexity and energy consumption can be achieved while an acceptable performance can still be maintained, which is especially useful for large array systems. In particular, we will use the quaternion-valued steering vector model for crossed-dipole arrays [2, 4, 11, 12, 13, 14, 15, 16, 17, 18], and propose a novel quaternion-valued adaptive algorithm for reference signal based beamforming.

Based on these recent advances in quaternion-valued signal processing, we here derive a reweighted zero attracting (RZA) quaternion-valued least mean square (QLMS) algorithm by introducing a RZA term to the cost function of the QLMS algorithm. Similar to the idea of the RZA least mean square (RZA-LMS) algorithm proposed in [19], the RZA term aims to have a closer approximation to the l_0 norm so that the number of non-zero valued coefficients can be reduced more effectively in the adaptive beamforming process. This algorithm can be considered as an extension of our recently proposed zero-attracting QLMS (ZA-QLMS) algorithm [69], where the l_1 norm penalty term was used in the update equation of the weight vector due to the sparsity-promoting nature of l_1 norm minimisation, and much of the recent research has employed that. We will show in our simulations that the RZA-LMS algorithm has a much better performance in terms of both steady state error and the number of sensors employed after convergence.

This chapter is organized as follows. A review of the complex-valued array model and beamforming structure is given in Section 4.1. The quaternion-based array signal model and the reference signal based adaptive beamforming structure is provided in Section 4.2, and the proposed RZA-QLMS algorithm is derived in Section 4.3. Simulation results are presented in Section 4.4, followed by a summary in Section 4.5.

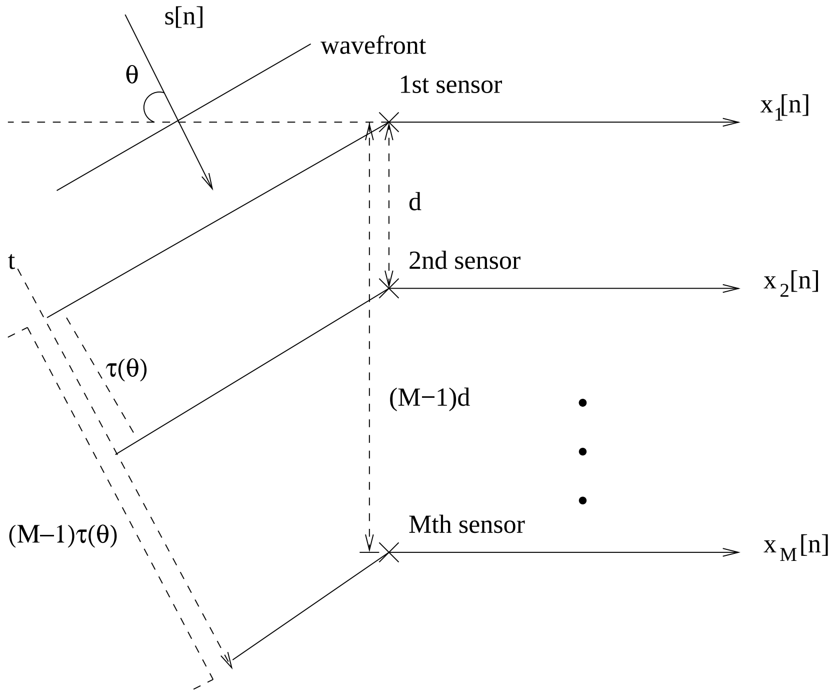

Fig. 4.1: A ULA with M omnidirectional sensors.

4.1 Introduction to Adaptive Beamforming

4.1.1 Array model and beamforming structure

Based on different spatial locations of the sensors, arrays can be categorised into three classes: linear arrays, planar arrays and volumetric arrays [22, 64]. Due to the simple structure and lower computational complexity in practical applications, linear arrays have been a focus of research since the very beginning of the array signal processing area. In our study, we will focus on the uniform linear arrays (ULAs).

A ULA with M omnidirectional sensors is shown in Fig. 4.1, and the distance between two adjacent sensors is d. The plane wave signal $s[n]$ arrives at the array from the far field. The direction of the signal is measured by an angle

θ with respect to the broadside, which is perpendicular to the line of sensors. The delay τ between two adjacent sensors for the input signal with known DOA angle θ is expressed as

$$\tau(\theta) = \frac{d}{c}\sin\theta = 2\pi\frac{d}{\omega_0\lambda_0}\sin\theta \tag{4.1}$$

where c is the propagation speed of the wavefront and λ_0 is the wavelength corresponding to the centre frequency ω_0 of input signal.

If we assume that the signal received by the first sensor is $s[n]$, then the following ones arrive at the sensors accordingly with $(m-1)\tau(\theta)$ seconds delay, where $m = 1, \cdots, M$. We can also find that the received signal at the mth sensor has a phase shift $e^{-j\omega_0(m-1)\tau(\theta)}$ because of the propagation delay. If we express these phase shifts on all sensors of the array, a so-called steering vector $\mathbf{a}(\theta)$ of the input signal from DOA angle θ will be obtained based on that

$$\mathbf{a}(\theta) = [1, e^{-j2\pi d\sin\theta/\lambda_0}, \cdots, e^{-j2\pi(M-1)d\sin\theta/\lambda_0}]^T \tag{4.2}$$

where the superscript $(\cdot)^T$ represent the transpose operation. Then, the received signals can be expressed as vector form

$$\mathbf{x}[n] = \mathbf{a}(\theta)s[n]. \tag{4.3}$$

For a general beamforming structure, the output of this array will be a mixture of all of the arrived signals including the interested signals as well as interferences. The purpose of beamforming is to select the desired signals from specific direction while suppressing the interferences from other directions, thereby adjusting the weight vector **w** to form a appropriate beampattern having a peak at the desired direction and nulls at the interferences directions. As to the basic beamformer structure, the received signals through sensors are always multiplied by a set of coefficient weight vectors and then summed up to form into the beamformer output $y[n]$

$$y[n] = \mathbf{w}[n]^T \mathbf{x}[n] \tag{4.4}$$

The beam pattern of a beamformer is its response to the impinging signals as a function of θ, which is given by

$$B(\theta) = \mathbf{w}^T \mathbf{a}(\theta), \tag{4.5}$$

where $\mathbf{a}(\theta)$ is the steering vector and the magnitude response $|B(\theta)|$ is adopted to describe the change of a beamformer with regards to the signal arriving from distinguished DOA angles.

Furthermore, as a beamformer has to track the changing signals in many practical scenarios, adaptive beamforming is required to obtain the data-dependent optimum output. Various adaptive beamforming methods have been proposed in the past decades, and one representative example is the reference signal based (RSB) beamformer [39, 70, 71, 72, 73].

4.1.2 LMS-based RSB beamformer

For a LMS-based RSB beamformer, the beamforming process can be regarded as a conventional adaptive filtering problem, which is similar to the standard adaptive filter structure introduced before. The general structure of such a beamformer is given in Fig. 4.2 [74, 75, 76].

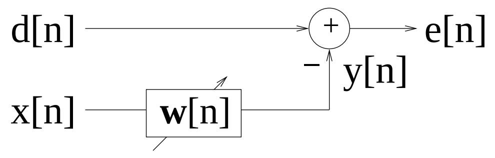

Fig. 4.2: Reference signal based beamforming structure.

The error signal $e[n]$ is generated by the difference between the reference signal and the beamformer output, which is given by

$$e[n] = d[n] - \mathbf{w}^H[n]\mathbf{x}[n], \tag{4.6}$$

where $d[n]$ is the reference signal, $\mathbf{w}[n]$ denotes the adaptive weight vector, $\mathbf{x}[n]$ is the input data vector, and $\{\cdot\}^T$ denotes the transpose operation. The cost function $J_2[n]$ is constructed by using the mean square error (MSE) $E\{|e[n]|^2\}$ and can be expressed as [77, 78]

$$\begin{aligned} J_2[n] &= E\{|e[n]|^2\} = E\{(d[n] - \mathbf{w}^H[n]\mathbf{x}[n])^2\} \\ &= E\{|d[n]|^2\} - \mathbf{w}^H[n]\mathbf{p} - \mathbf{p}^H\mathbf{w}[n] + E\{\mathbf{w}^H[n]x[n]x^H[n]\mathbf{w}[n]\} \end{aligned} \tag{4.7}$$

where $\mathbf{p} = E\{x[n]d^*[n]\}$. Then the gradient $\nabla J_2[n]$ is expressed as

$$\nabla J_2[n] = \frac{\partial J_2[n]}{\partial \mathbf{w}^*}. \tag{4.8}$$

Finally, the update equation of LMS-based adaptive algorithm with step size μ_2 is expressed as

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu_2 e^*[n]\mathbf{x}[n]. \tag{4.9}$$

4.2 Adaptive Beamforming Based on Vector Sensor Arrays

4.2.1 Quaternionic array signal model

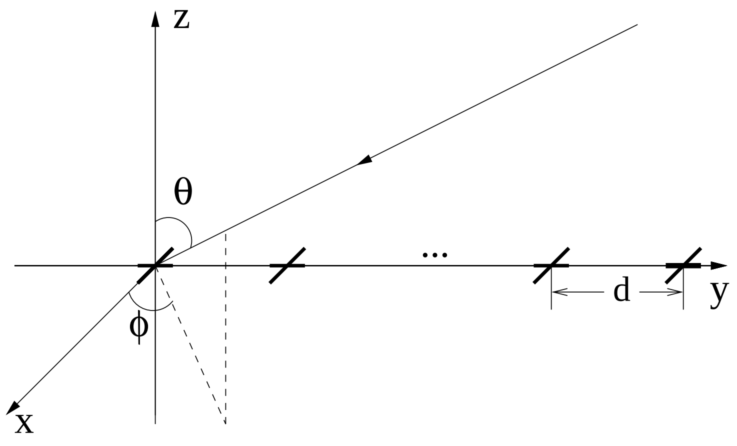

Fig. 4.3: A ULA with crossed-dipoles.

A general structure for a uniform linear array (ULA) with M crossed-dipole pairs is shown in Fig. 4.3, where these pairs are located along the y-axis with an adjacent distance d, and at each location the two crossed components are parallel to the x-axis and y-axis, respectively. For a far-field incident signal with a direction of arrival (DOA) defined by the angles θ and ϕ, its spatial steering vector is given by

$$\begin{aligned}\mathbf{S}_c(\theta,\phi) &= [1, e^{-j2\pi d\sin\theta\sin\phi/\lambda}, \\ &\quad \cdots, e^{-j2\pi(M-1)d\sin\theta\sin\phi/\lambda}]^T \end{aligned} \tag{4.10}$$

where λ is the wavelength of the incident signal and $\{\cdot\}^T$ denotes the transpose operation. For a crossed dipole the spatial-polarization coherent vector can be given by [79, 80]

$$\mathbf{S}_p(\theta,\phi,\gamma,\eta) = \begin{cases} [-\cos\gamma, \cos\theta\sin\gamma e^{j\eta}] & \text{for } \phi = \pi/2 \\ [\cos\gamma, -\cos\theta\sin\gamma e^{j\eta}] & \text{for } \phi = -\pi/2 \end{cases} \tag{4.11}$$

where γ is the auxiliary polarization angle with $\gamma \in [0, \pi/2]$, and $\eta \in [-\pi, \pi]$ is the polarization phase difference.

The array structure can be divided into two sub-arrays: one parallel to the x-axis and one to the y-axis. The complex-valued steering vector of the x-axis sub-array is given by

$$\mathbf{S}_x(\theta,\phi,\gamma,\eta) = \begin{cases} -\cos\gamma\mathbf{S}_c(\theta,\phi) & \text{for } \phi = \pi/2 \\ \cos\gamma\mathbf{S}_c(\theta,\phi) & \text{for } \phi = -\pi/2 \end{cases} \tag{4.12}$$

and for the y-axis it is expressed as

$$\mathbf{S}_y(\theta,\phi,\gamma,\eta) = \begin{cases} \cos\theta\sin\gamma e^{j\eta}\mathbf{S}_c(\theta,\phi) & \text{for } \phi = \pi/2 \\ -\cos\theta\sin\gamma e^{j\eta}\mathbf{S}_c(\theta,\phi) & \text{for } \phi = -\pi/2 \end{cases} \tag{4.13}$$

Combining the two complex-valued subarray steering vectors together, an overall quaternion-valued steering vector with one real part and three imaginary parts can be constructed as

$$\begin{aligned} \mathbf{S}_q(\theta,\phi,\gamma,\eta) &= \Re\{\mathbf{S}_x(\theta,\phi,\gamma,\eta)\} + i\Re\{\mathbf{S}_y(\theta,\phi,\gamma,\eta)\} + \\ & \quad j\Im\{\mathbf{S}_x(\theta,\phi,\gamma,\eta)\} + k\Im\{\mathbf{S}_y(\theta,\phi,\gamma,\eta)\}, \end{aligned} \tag{4.14}$$

where $\Re\{\cdot\}$ and $\Im\{\cdot\}$ are the real and imaginary parts of a complex number/vector, respectively. Given a set of coefficients, the response of the array

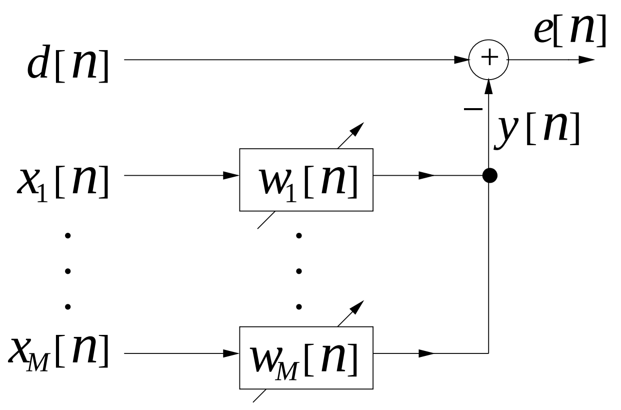

Fig. 4.4: Reference signal based adaptive beamforming.

is given by

$$r(\theta, \phi, \gamma, \eta) = \mathbf{w}^H \mathbf{S}_q(\theta, \phi, \gamma, \eta) \tag{4.15}$$

where **w** is the quaternion-valued weight vector.

4.2.2 Reference signal based quaternion-valued adaptive beamforming

The reference signal based beamformer is a very important class of adaptive beamformers, and one particular application is the smart antenna technique in wireless communications, where a reference signal is usually available. As just mentioned, when a reference signal $d[n]$ is available, adaptive beamforming can be implemented by the standard adaptive filtering structure, as shown in Fig. 4.4, where $x_m[n]$, $m = 1, \cdots, M$ are the received quaternion-valued input signals through the M pairs of crossed-dipoles, and $w_m[n] = a_m + b_m i + c_m j + d_m k$, $m = 1, \cdots, M$ are the corresponding quaternion-valued weight coefficients with a, b, c and d being real-valued. $y[n]$ is the beamformer output and $e[n]$ is the error signal

$$\begin{aligned} y[n] &= \mathbf{w}^T[n]\mathbf{x}[n] \\ e[n] &= d[n] - \mathbf{w}^T[n]\mathbf{x}[n] \ , \end{aligned} \tag{4.16}$$

where

$$\begin{aligned} \mathbf{w}[n] &= [w_1[n], w_2[n], \cdots, w_M[n]]^T \\ \mathbf{x}[n] &= [x_1[n], x_2[n], \cdots, x_M[n]]^T \ . \end{aligned} \tag{4.17}$$

The conjugate form of the error signal is $e^*[n]$, given by

$$e^*[n] = d^*[n] - \mathbf{x}^H[n]\mathbf{w}^*[n], \tag{4.18}$$

where $\{\cdot\}^H$ is the combination of both $\{\cdot\}^T$ and $\{\cdot\}^*$ operations for a quaternion. Then **w** can be updated by minimizing the instantaneous square error.

As discussed in [54, 81], the gradient of $J_3[n]$ with respect to $\mathbf{w}^*$ would give the steepest direction for the optimization surface. It can be obtained as follows

$$\nabla_{\mathbf{w}^*} J_3[n] = -\frac{1}{2} e[n]\mathbf{x}^*[n] \ , \tag{4.19}$$

and the update equation for the weight vector with step size μ_3 is given by

$$\mathbf{w}[n+1] = \mathbf{w}[n] - \mu_3 \nabla_{\mathbf{w}^*} J_0[n], \tag{4.20}$$

leading to the following QLMS algorithm [10, 16, 82]

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \frac{1}{2}\mu_3 (e[n]\mathbf{x}^*[n]). \tag{4.21}$$

4.3 The RZA-QLMS Algorithm

Using the above adaptive QLMS algorithm, we can find the optimal coefficient vector in terms of minimum mean square error (MSE) and obtain a satisfactory beamforming result. However, to reduce the complexity and also power consumption of the system, in particular for a large array, we can reduce the number of sensors involved in the adaptive beamforming process, at the cost of the beamforming performance. To achieve this, we here derive a novel quaternion-valued adaptive algorithm by introducing an RZA term to the original cost function of the QLMS algorithm. In this way, we can simultaneously minimise the number of sensors involved while suppressing the interferences during the beamforming process.

First, to minimise the number of sensors, we could add the l_0 norm of the weight vector **w** to the cost function $J_3[n]$ to form a new cost function

$$\hat{J}_3[n] = (1-\delta_1)e[n]e^*[n] + \delta_1 \parallel \mathbf{w}[n] \parallel_0, \tag{4.22}$$

where δ_1 is a weighting term between the original cost function and the newly introduced term. In this way, the number of non-zero valued coefficients in **w** will be minimised too, where a similar idea has been applied in [83].

In practice, we could replace the l_0 norm by the l_1 norm. However, l_1 norm would uniformly penalise all non-zero valued coefficients, while l_0 norm penalises smaller non-zero values more heavily than those larger non-zero values. To have a closer approximation to l_0 norm, we can introduce a larger weighting term to those coefficients with smaller values and a smaller weighting term to those with larger values. This weighting term will change according to the resultant coefficients at each update of the algorithm. This general idea has been implemented as a reweighted l_1 minimization [84, 85, 86] and employed in the sparse array design problem [87, 88, 89].

The modified cost function for the proposed RZA-QLMS algorithm with the reweighting term is given by

$$J_6[n] = (1-\delta_1)e[n]e^*[n] + \delta_1 \sum_{m=1}^{M}(\varepsilon_m|w_m[n]|), \tag{4.23}$$

where ε_m is the reweighting term for w_m. In deriving the gradient, we have treated the weighting term as a constant as an approximation. This has been a widely adapted practice to achieve a trade-off between complexity and performance in reweighted l_1 norm approximation. This is also similar to the practice in the derivation of many variable step size adaptive filtering algorithms, where the variation in step size has been added at a later stage, after derivation of the gradient. Then using the chain rule in [81], we can obtain the gradient of the above cost function $J_6[n]$ with respect to $\mathbf{w}^*[n]$. In particular, the differentiation of the second part of the cost function with regards to $w_m^*[n]$ is given by

$$\begin{aligned}
&\frac{\partial(\varepsilon_m|w_m[n]|)}{\partial w_m^*}\\
=\;& \frac{1}{4}\varepsilon_m(\frac{\partial(|w_m[n]|)}{\partial a_m} + \frac{\partial(|w_m[n]|)}{\partial b_m}i\\
&+\frac{\partial(|w_m[n]|)}{\partial c_m}j + \frac{\partial(|w_m[n]|)}{\partial d_m}k)\\
=\;& \frac{1}{4}\varepsilon_m(\frac{a_m}{|w_m[n]|} + \frac{b_m}{|w_m[n]|}i + \frac{c_m}{|w_m[n]|}j + \frac{d_m}{|w_m[n]|}k)\\
=\;& \frac{1}{4}\varepsilon_m\frac{w_m[n]}{|w_m[n]|} = \frac{1}{4}\varepsilon_m(sign(w_m[n]))\ ,
\end{aligned} \tag{4.24}$$

where $sign(\cdot)$ is a component-wise sign function

$$sign(w_m[n]) = \begin{cases} w_m[n]/|w_m[n]| & w_m[n] \neq 0 \\ 0 & w_m[n] = 0 \end{cases}$$

The overall gradient result is given by

$$\nabla_{w_m^*} J_6[n] = -\frac{1}{2}(1-\delta_1)e[n]x_m^*[n] + \frac{1}{4}\delta_1 \varepsilon_m (sign(w_m[n])). \tag{4.25}$$

We choose the reweighting term ε_m as

$$\varepsilon_m = 1/(\sigma + |w_m[n]|), \tag{4.26}$$

with σ being roughly the threshold value below which the corresponding sensor will not be included in the update.

This can be understood in this way: when the coefficient value is much larger than σ, $\frac{|w_m[n]|}{\sigma+|w_m[n]|}$ will be almost 1 and the reweighted norm will be almost the same as the l_0 norm; when the coefficient value is much smaller than σ, $\frac{|w_m[n]|}{\sigma+|w_m[n]|}$ will be a very small value and it will not contribute sufficiently to the reweighted l_1 norm, i.e. we do not minimise that coefficient any more and it can be considered to be zero. As a result, the σ should be small enough.

Then, with the step size μ_6, we finally obtain the following update equation for the proposed RZA-QLMS algorithm in vector form

$$\begin{aligned} \mathbf{w}[n+1] &= \mathbf{w}[n] + \frac{1}{2}(\mu_6 - 4\rho_2)(e[n]\mathbf{x}^*[n]) \\ &\quad -\rho_2(sign(\mathbf{w}[n]))./(\sigma + |\mathbf{w}[n]|)\ , \end{aligned} \tag{4.27}$$

where $\rho_2 = \frac{1}{4}\mu_6\delta_1$, $|\mathbf{w}[n]|$ is a vector formed by taking the absolute value of the coefficients in $\mathbf{w}[n]$, './' is a component-wise division between two vectors, and

Table 4.1: Comparison of computational complexity.

	QLMS	ZA-QLMS	RZA-QLMS
Real-valued addition	28M+4	35M+4	38M+4
Real-valued multiplication	32M+4	44M+4	52M+4
Square root operation	0	M	2M

$sign(\mathbf{w}[n])$ is defined as

$$sign(\mathbf{w}[n]) = \begin{cases} \mathbf{w}[n]./|\mathbf{w}[n]| & \mathbf{w}[n] \neq 0 \\ 0 & \mathbf{w}[n] = 0 \end{cases}$$

When $\sigma + |\mathbf{w}[n]|$ is removed from the above equation, it will be reduced to the ZA-QLMS algorithm in [69], with its cost function given by

$$J_7[n] = (1 - \delta_2)e[n]e^*[n] + \delta_2 \|\mathbf{w}[n]\|_1 \ , \tag{4.28}$$

where δ_2 is a trade-off factor. The update equation for the ZA-QLMS algorithm is

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \frac{1}{2}(\mu_7 - 4\rho_3)(e[n]\mathbf{x}^*[n]) - \rho_3 \cdot sign(\mathbf{w}[n]) \ , \tag{4.29}$$

where $\rho_3 = \frac{1}{4}\mu_7\delta_2$, and μ_7 is the step size.

We now discuss the computational complexity of the algorithms. The results are shown in Tab. 4.1, where M is the number of vector sensors of the array. Obviously, the proposed RZA-QLMS algorithm has the highest computational complexity. However, as we will see in simulations, this additional cost is paid back by a resultant much smaller number of sensors and especially at a later stage of the adaptation, when the number of sensors involved in the update becomes smaller, the overall complexity of the RZA-QLMS algorithm could be lower than the other two algorithms.

After removing the sensors with a smaller magnitude for their coefficients compared to σ, the beam response difference Δr between the original array and the new one is given by

$$\begin{aligned}\Delta r &= |\mathbf{w}^H\mathbf{S}_q - (\mathbf{w}-\Delta\mathbf{w})^H\mathbf{S}_q| \\ &= |\Delta\mathbf{w}^H\mathbf{S}_q| \leq |\Delta\mathbf{w}^H|\cdot|\mathbf{S}_q| \leq \sigma\cdot\Delta M\cdot\sqrt{M} \qquad (4.30)\end{aligned}$$

where ΔM is the number of removed sensors, and $\Delta\mathbf{w}$ is the change of $\mathbf{w}$ after some of its sensors are removed (these corresponding coefficients on the positions of removed sensors have a magnitude smaller than σ). As a result, the maximum possible change in array response, due to removal of some sensors, is given by $\sigma\cdot\Delta M\cdot\sqrt{M}$.

4.4 Simulation Results

Simulations are performed based on a vector sensor array with 16 crossed-dipoles and half-wavelength spacing for the three algorithms: QLMS, ZA-QLMS and RZA-QLMS. The stepsizes μ, μ_1 and μ_2 are set to be 2×10^{-4}, 4×10^{-4} and 2×10^{-4}, respectively, which are chosen empirically to make sure these algorithms have a similar convergence speed. A desired signal with 20 dB signal to noise ratio (SNR) impinges from the broadside of the array ($\theta=0^\circ$) and two interfering signals with a signal to interference ratio (SIR) of -10 dB arrive from the directions $(20^\circ, 90^\circ)$, and $(30^\circ, -90^\circ)$, respectively. All

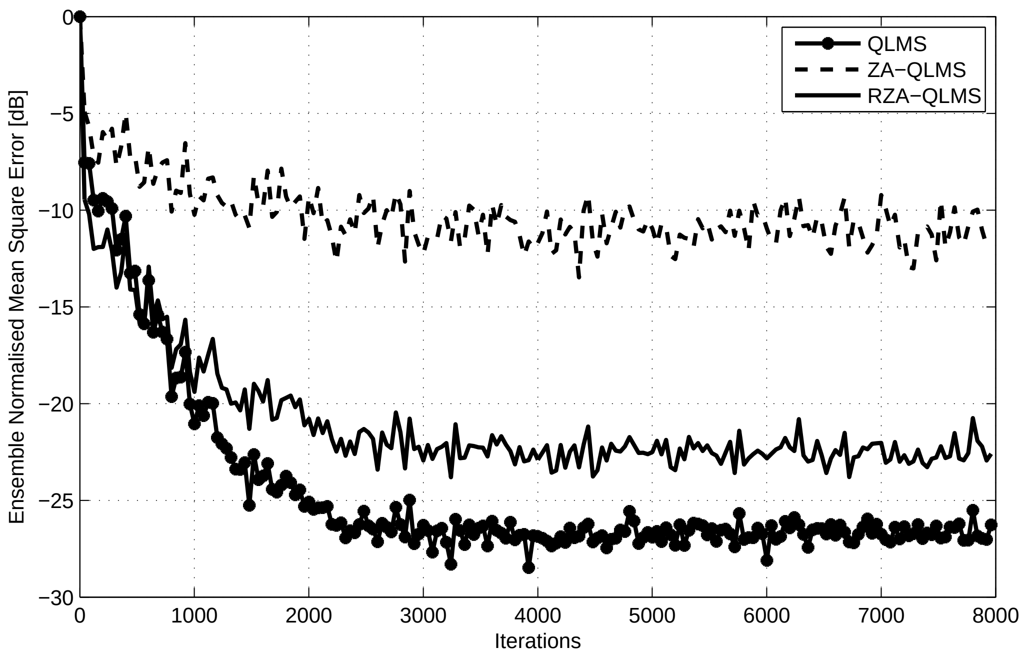

Fig. 4.5: Learning curves of the three algorithms.

the signals have the same polarisation of $(\gamma, \eta) = (30^\circ, 0)$. For the RZA-QLMS and ZA-QLMS algorithms, the coefficients of the zero attractor ρ_1 and ρ_2 are 7×10^{-7} and 2.8×10^{-5}, respectively and $\sigma = 0.001$. Their learning curves obtained by averaging results from 200 simulation runs are shown in Fig. 4.5 and the resultant beam patterns are shown in Fig. 4.6, where for convenience positive values of θ indicate the value range $\theta \in [0^\circ,\ 90^\circ]$ for $\phi = 90^\circ$, while negative values of $\theta \in [-90^\circ,\ 0^\circ]$ indicate an equivalent range of $\theta \in [0^\circ,\ 90^\circ]$ with $\phi = -90^\circ$.

First, the two nulls at the directions of the interfering signals can be observed in all three beam patterns, clearly demonstrating that all of the algorithms have achieved a satisfactory beamforming result. However, from Fig. 4.5, we see that although these three algorithms have a similar convergence speed, the original

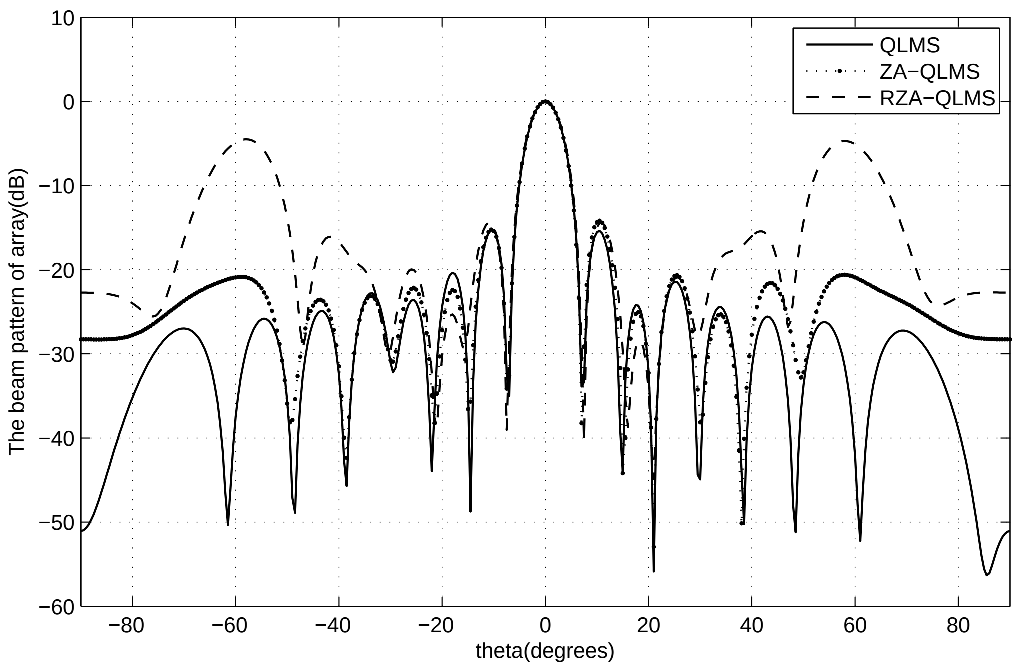

Fig. 4.6: Beam patterns of the three algorithms with 0° desired signal.

QLMS algorithm has the smallest steady state error, which is not surprising since it has the most degrees of freedom among them. On the other hand, the proposed RZA-QLMS algorithm has achieved a lower steady state error than the ZA-QLMS algorithm.

In terms of output signal to interference plus noise ratio (SINR), it is 23.48 dB for the QLMS algorithm, 18.32 dB for the RZA-QLMS algorithm, and 7.36 dB for the ZA-QLMS algorithm. The output SINR is obtained by finding the power of the desired signal at the output of the beamformer and also the total power of noise and interferences at the output and then calculate the ratio between the two. In our simulations, we have all the signals (desired, interference and noise). After obtaining the optimum weight vector, we then pass only the desired signal through the system and the power of the output signal will be

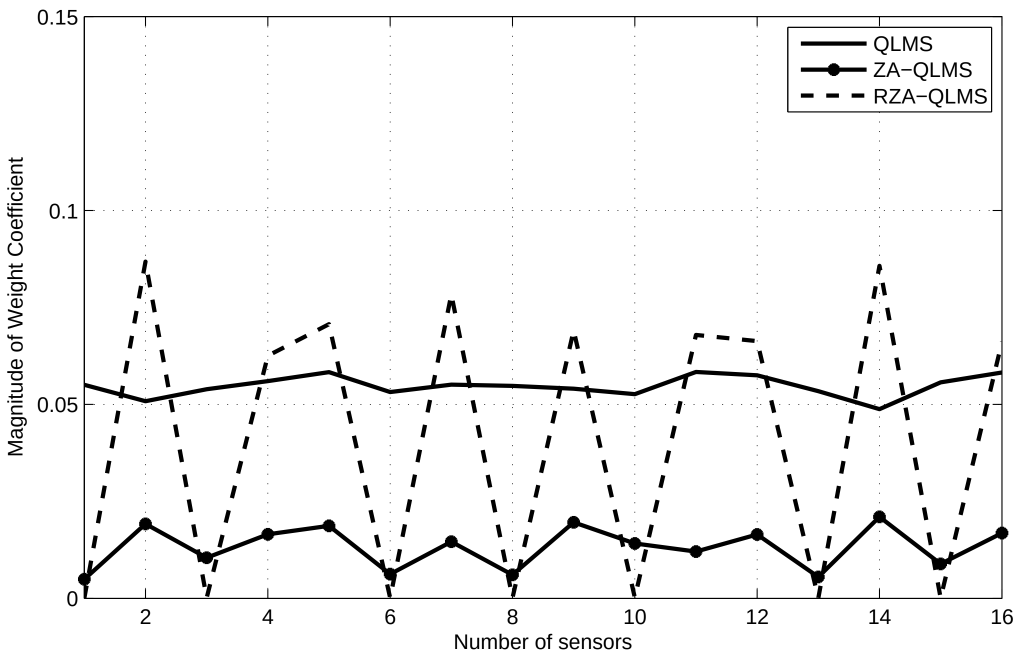

Fig. 4.7: Amplitudes of the steady state weight coefficients.

the power of the desired signal at the output of the beamformer; similarly, we pass only the interferences and noise through the system and the power of the output signal in this case will be the power of the interferences plus noise at the beamformer output.

The reason for the higher sidelobes of the RZA-QLMS algorithm is, by imposing zeros effectively to the system, the number of available sensors will be smaller than the other two algorithms and with a smaller sensor number, the sidelobe will increase naturally. However, we can not simply evaluate the performance of the different algorithms based on the sidelobe levels, as this is an adaptive beamforming system and the response of the array at the directions of interfering signals plays a much more important role in determining the output SINR.

Table 4.2: Comparison of computational complexity with sensor numbers

	QLMS	ZA-QLMS	RZA-QLMS
Real-valued addition	452	564	346
Real-valued multiplication	516	708	472
Square root operation	0	16	18

The amplitudes of steady state weight coefficients for the three algorithms are shown in Fig. 4.7, where for the QLMS algorithm, the amplitudes of the coefficients are spread over the sixteen sensors with some small variations, while for the ZA-QLMS algorithm, as the l_1 norm of the weight vector is minimised, some degree of sparsity has also been achieved with four of the coefficients are close to zero. However, with 0.001 as the threshold value, they can not be discarded. For the RZA-QLMS algorithm, the variation is significantly larger and seven of them are almost zero-valued, which means the corresponding sensors can be removed and only 9 sensors are needed to give a satisfactory beamforming result, rather than 16 sensors. Moreover, the difference response between the original array and the one with 7 sensors removed is small, and no difference can really be observed by a naked eye, as shown in Fig. 4.8.

Based on the steady-state sensor number, the computational complexity of the three algorithms is listed in Tab. 4.2, where we can see that the RZA-QLMS algorithm has the lowest complexity.

For the direction of the desired signal, we can consider any other directions and the proposed algorithms are not limited to the zero degree case. For example, we can consider the case of the desired signal coming from the direction 15°

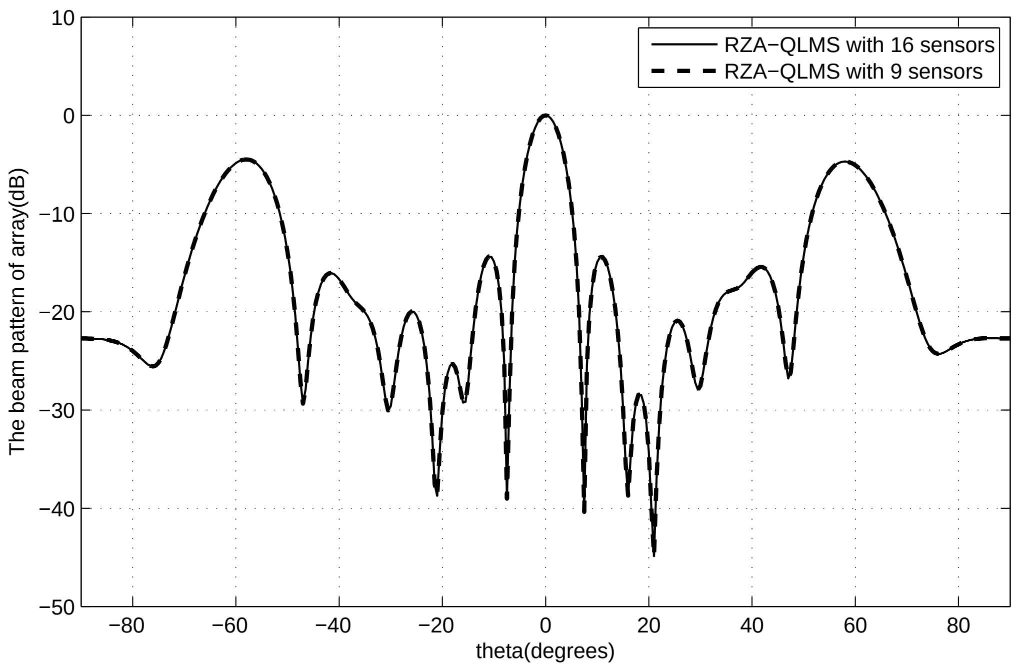

Fig. 4.8: Beam pattern of the two arrays.

and the resultant beam pattern is shown in Fig. 4.9 in this reply, where the two interfering signals come from 30° and -15°, respectively. We can see from the figure that the algorithms have worked well by forming a beam to the desired direction and two nulls at the interference directions.

4.5 Summary

A brief review of adaptive beamforming was provided first and then a reweighted zero attracting quaternion-valued least mean square algorithm was derived for reference signal based adaptive beamforming based on vector sensor arrays consisting of crossed dipoles. It can reduce the number of sensors involved in the beamforming process so that reduced system complexity and energy con-

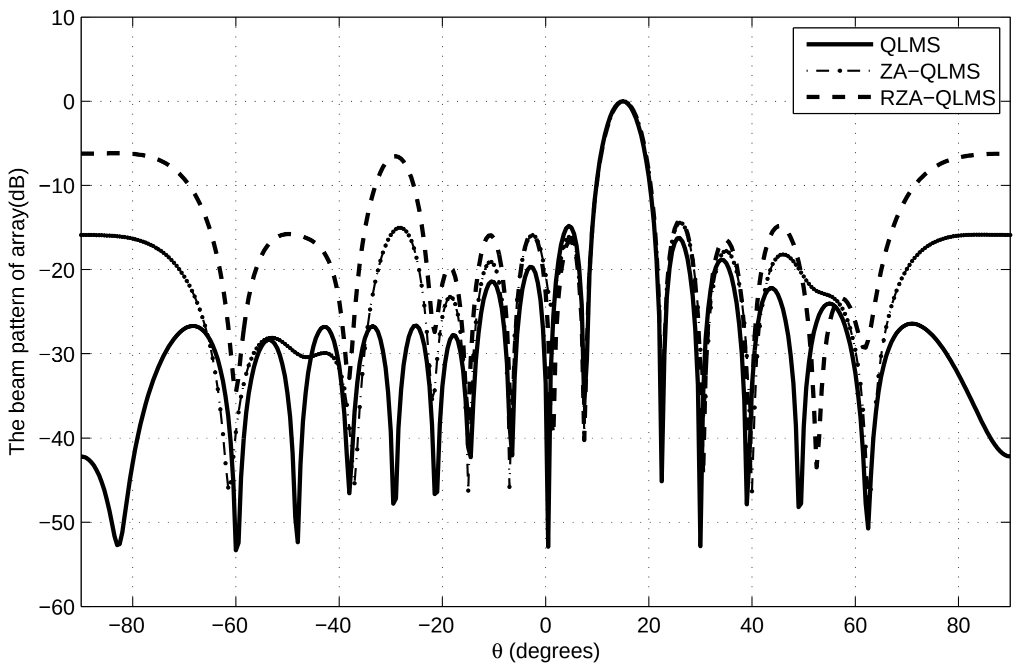

Fig. 4.9: Beam patterns of the three algorithms with 15° desired signal.

sumption can be achieved while an acceptable performance can still be maintained, which is especially useful for large array systems. Simulation results have shown that the proposed algorithm can work effectively for beamforming while enforcing a sparse solution for the weight vector where the corresponding crossed-dipole sensors with zero-valued coefficients can be removed from the system.

Chapter 5

Application to Wind Profile Prediction

Wind profile (including speed and direction) prediction at different scales (short-term, mid-term and long-term) plays a crucial role for efficient operation of wind turbines and wind power prediction. In this study, the wind profile is related to three dimensional wind speed prediction in continuous time series. This problem can be approached in two different ways: one is based on statistical signal processing techniques and both linear and nonlinear (such as artificial neural networks) models can be employed either separately or combined together for profile prediction; on the other hand, wind/atmospheric flow analysis is a classical problem in computational fluid dynamics (CFD) in applied mathematics, which employs various numerical methods and algorithms, although it is an extremely time-consuming process with high computational complexity. In this chapter, we will focus on the application of the signal processing approach to wind profile prediction and a combination of the signal processing approach and the CFD approach will be studied in the next chapter.

Here we will first perform a correlation analysis to the wind flow data generated by CFD, by using the Wiener solution to find out the optimum weight vector and then calculate the minimum prediction error using the correlation distribution of the data samples. Obviously, the correlations and then the prediction errors will be different with different sampling frequencies and prediction advance steps. The aim here is to obtain a smaller prediction error and a longer prediction advance time by finding the appropriate sampling frequency, since a higher sampling frequency will increase the computational complexity, whereas a lower sampling frequency will lead to a larger prediction error. Then we will propose a multi-channel frequency-domain QLMS algorithm and apply it to the wind profile prediction problem.

This chapter is organised as follows. Correlation analysis is presented in Section 5.1, while the frequency-domain multi-channel quaternion-valued adaptive filtering problem will be studied in Section 5.2 . Simulation results are given in Section 5.3 using the CFD generated data and also the recorded wind data from multiple measurement sites. Section 5.4 briefly summarises the chapter.

5.1 Correlation Analysis

5.1.1 The Wiener solution

For the signal processing based approach, we can use the Wiener solution for quasi-stationary scenarios and the QLMS algorithm for the more general case with constantly changing data statistics. Here we introduce the Wiener solution first.

To obtain the Wiener solution, we assume a reference signal $d[n]$ is available, which would be the most up to date measurement of the wind speed in our case. Then we can work out the prediction error $e[n]$ after finding out the optimum weight vector $\mathbf{w}[n]$.

We use $s[n]$ to represent the input data sequence to the predictor. The error signal $e[n]$ is given by

$$e[n] = d[n] - \mathbf{w}^T[n]\mathbf{s}[n], \tag{5.1}$$

where $\mathbf{w}[n] = [w_1[n], w_2[n], \cdots, w_L[n]]^T$ is the weight vector with length L and $\mathbf{s}[n] = [s[n], s[n-1], \cdots, s[n-L+1]]^T$ is the corresponding L-by-1 input data vector. From [31], based on the instantaneous gradient result, the optimum solution $\mathbf{w}_{opt}$ should satisfy

$$E\{-\frac{1}{2}\mathbf{s}[n]e^*[n]\} = 0. \tag{5.2}$$

In particular,

$$\begin{aligned} E\{\mathbf{s}[n]e^*[n]\} &= E\{\mathbf{s}[n]d^*[n]\} - E\{\mathbf{s}[n]\mathbf{s}[n]^H\mathbf{w}^*_{opt}\} \\ &= \mathbf{p} - \mathbf{R}_s\mathbf{w}^*_{opt} = 0, \end{aligned} \tag{5.3}$$

where the cross-correlation vector $\mathbf{p} = E\{\mathbf{s}[n]d^*[n]\}$ and the covariance matrix $\mathbf{R}_s = E\{\mathbf{s}[n]\mathbf{s}[n]^H\}$. Note that the superscript H denotes Hermitian transpose. Then, we have

$$\mathbf{w}^*_{opt} = \mathbf{R}_s^{-1}\mathbf{p}. \tag{5.4}$$

As shown, the optimum weight vector can be obtained by the above equation directly. For the covariance matrix, $\mathbf{R}_s$ can be expressed in expanded form using the condition of wide-sense stationarity, and we have

$$\mathbf{R}_s = \begin{bmatrix} r(0) & r(1) & \cdots & r(L-1) \\ r^*(1) & r(0) & \cdots & r(L-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(L-1) & r^*(L-2) & \cdots & r(0) \end{bmatrix} \tag{5.5}$$

where the $r(k)$,$k = 0, 1, \cdots, L-1$ is the autocorrelation function of the process for a lag of k given by

$$r(k) = E[s[n]s^*[n-k]],\ k = 0, 1, \cdots, L-1. \tag{5.6}$$

Correspondingly, the cross-correlation vector **p** can be described by the expanded form below

$$\mathbf{p} = [p(0), p(-1), \cdots, p(1-L)]^T. \tag{5.7}$$

where $p(-k) = E[s(n)d^*[n]]$, $k = 0, 1, 2, \cdots, L-1$.

5.1.2 Correlations with different sampling frequencies

As mentioned, an appropriate sampling frequency is essential for effective and efficient prediction. Several data sets with different sampling frequencies will be used to work out the normalised error and then compare them to find out the desired sampling frequency for the studied scenario. The data is generated by a CFD software employing the Direct Numerical Simulation (DNS) method. For each sampling frequency, a sample sequence will be produced for calculating the correlation matrix/vector to obtain the Wiener solution. Then, we can

Table 5.1: Normalised prediction error at sampling frequency f_s = 5Hz with prediction step P and number of samples involved N (Part I).

	N=1-1000	N=1-1500	N=1-2000	N=1-2500	N=1-3000
P=1	0.0039	0.0057	0.0051	0.0048	0.0048
P=2	0.0315	0.0393	0.037	0.035	0.035
P=3	0.0920	0.1060	0.1010	0.0990	0.0990
P=4	0.1777	0.1956	0.1863	0.1830	0.1844
P=5	0.2748	0.2922	0.2793	0.2758	0.2773
P=6	0.3728	0.3865	0.3706	0.3675	0.3683

find the proper sampling frequency with reasonable range of errors and use this sampling frequency for signal processing based prediction.

5.1.3 Simulation results

The data we used is generated by a CFD software employing the DNS method. We have chose 9 points from the periodic boxes with 262144 grids. First, the sampling frequency is set as $f_s = 5Hz$, i.e., with a sampling interval of 0.2s. We use the Wiener solution to find out the optimum weight vector and then calculate the minimum prediction error using the correlation distribution of the data samples. The results for different sampling frequencies are listed in Tables 5.1-5.2, where the column direction is the prediction advance step from P=1 to P=6, and the row direction is the number of samples used for obtaining the correlation matrix/vector to calculate the Wiener solution. The values in the table are the resultant prediction error normalised by the power of corresponding data sequence.

Table 5.2: Normalised prediction error at sampling frequency f_s = 5Hz with prediction step P and number of samples involved N (Part II).

	N=1-3500	N=1-4000	N=1-4500	N=1-5000	N=1-10000	N=1-20000
P=1	0.0056	0.0054	0.0053	0.0052	0.0048	0.0050
P=2	0.038	0.038	0.038	0.037	0.0359	0.0374
P=3	0.1030	0.1030	0.1030	0.1030	0.1008	0.1040
P=4	0.1886	0.1897	0.1901	0.1897	0.1859	0.1904
P=5	0.2805	0.2826	0.2838	0.2845	0.2774	0.2826
P=6	0.3702	0.3729	0.3751	0.3772	0.3666	0.3719

Table 5.3: Normalised prediction error at sampling frequency f_s = 2Hz with prediction step P and number of samples involved N (Part I).

	N=1-500	N=1-1000	N=1-1500	N=1-2000	N=1-2500
P=1	0.1123	0.1206	0.1134	0.1069	0.1101
P=2	0.3639	0.3758	0.3516	0.3385	0.3431
P=3	0.5768	0.5868	0.5583	0.5431	0.5465
P=4	0.7247	0.7392	0.7140	0.6972	0.6987
P=5	0.8251	0.8424	0.8231	0.8074	0.8072

Table 5.4: Normalised prediction error at sampling frequency f_s = 2Hz with prediction step P and number of samples involved N (Part II).

	N=1-3000	N=1-3500	N=1-4000	N=1-4500	N=1-5000
P=1	0.1097	0.1071	0.1070	0.1063	0.1080
P=2	0.3448	0.3400	0.3422	0.3415	0.3452
P=3	0.5490	0.5458	0.5476	0.5496	0.5541
P=4	0.6993	0.6971	0.6978	0.7028	0.7065
P=5	0.8064	0.8037	0.8029	0.8088	0.8127

Table 5.5: Normalised prediction error at sampling frequency f_s = 1Hz with prediction step P and number of samples involved N (Part I).

	N=1-500	N=1-1000	N=1-1500	N=1-2000	N=1-2500
P=1	0.3595	0.3662	0.3549	0.3561	0.3533
P=2	0.7045	0.7072	0.6922	0.6904	0.6836
P=3	0.9050	0.8890	0.8705	0.8643	0.8542

Table 5.6: Normalised prediction error at sampling frequency f_s = 1Hz with prediction step P and number of samples involved N (Part II)

	N=1-3000	N=1-3500	N=1-4000	N=1-4500	N=1-5000
P=1	0.3619	0.3629	0.3629	0.3616	0.3614
P=2	0.6944	0.6944	0.7037	0.7036	0.7042
P=3	0.8619	0.8683	0.8721	0.8723	0.8725

We have also tried two other scenarios, with a sampled frequency of 2Hz and 1Hz, and the results are shown in Tables 5.3, 5.4, 5.5 and 5.6. We can see that the values shown in Tables 5.3 and 5.4 are larger than the case with the 5Hz sampling frequency given the same sample intervals. For example, the normalised error is around 0.3 at P=6 (the acceptable range) in the first case, whereas the error is around 0.3 for the second case when prediction step is P=2, which means the correlation is smaller when sampling frequency is decreased, as expected.

For the third case (sampling frequency is 1Hz), the errors are much larger than the previous two cases, even with a prediction step of P=2. Therefore, we have to choose a smaller prediction step in the simulation to maintain the reasonable range of error. This can also be found in Fig. 5.1 clearly, where the

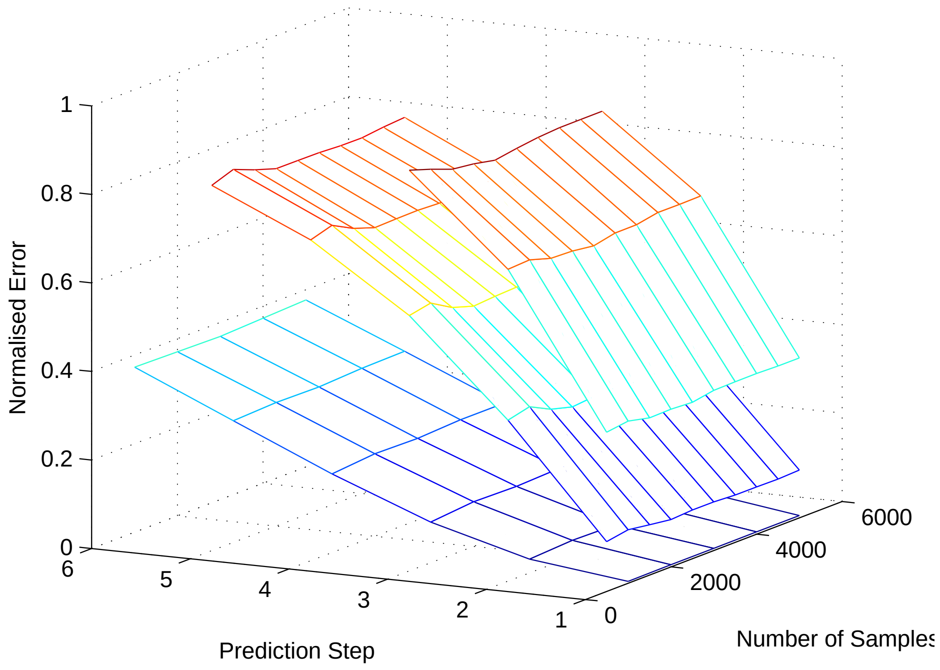

Fig. 5.1: Normalised error of wind prediction at different sample intervals.

bottom one is the error surface at 5Hz, and the top one is for 1Hz, while the middle one is for the 2Hz case.

As a result, we can see from the Fig. 5.2 that the normalised prediction error has increased as the value of prediction step P increases or the sampling frequency f_s decreases. This trend stays the same even if a larger number of data samples are used. Therefore, we can make prediction further in advance with smaller error when the correlation value is large, and the acceptable prediction step value depends on the correlation of these samples. On the other hand, we can estimate the prediction advance time through analysing the correlation distribution of the generated data.

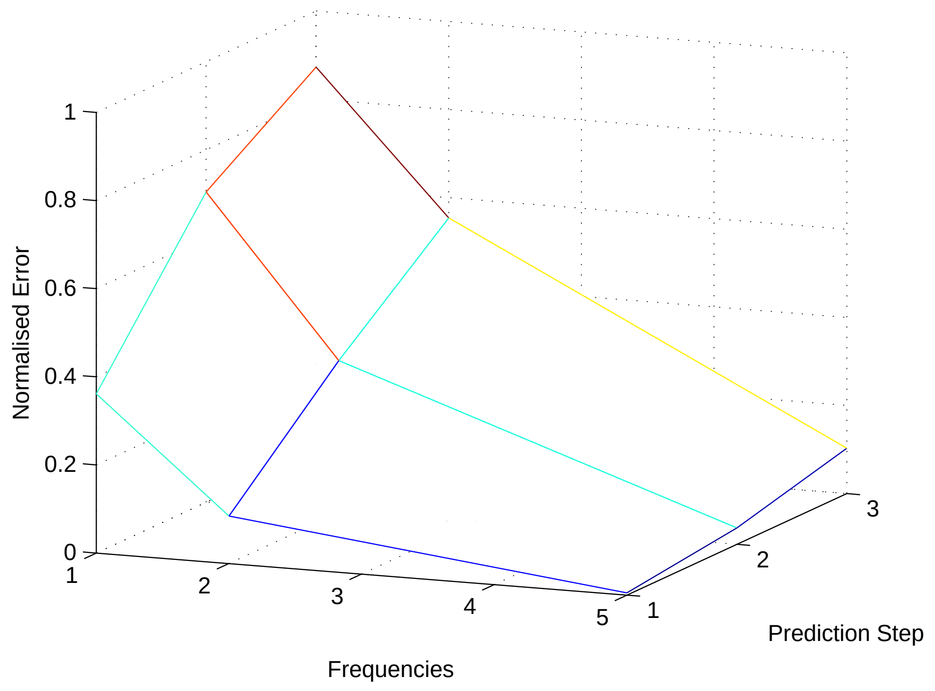

Fig. 5.2: Normalised error of wind prediction at different frequencies.

5.2 Transform-domain Quaternion-valued Adaptive Filtering and Its Application for Wind Profile Prediction

To increase the convergence speed and reduce the computational complexity of an adaptive filtering algorithm, traditionally we can transform the signals into the transform domain (also called frequency domain) or employing the more general subband techniques to split the signals into subbands, leading to frequency-domain or subband adaptive filtering structures [22, 90, 91, 92, 93]. With the introduction of quaternion-valued discrete Fourier transform (QDFT) and filter banks systems [26, 27], we can apply the same approach to quaternion-valued adaptive algorithms with similar advantages as in the complex domain.

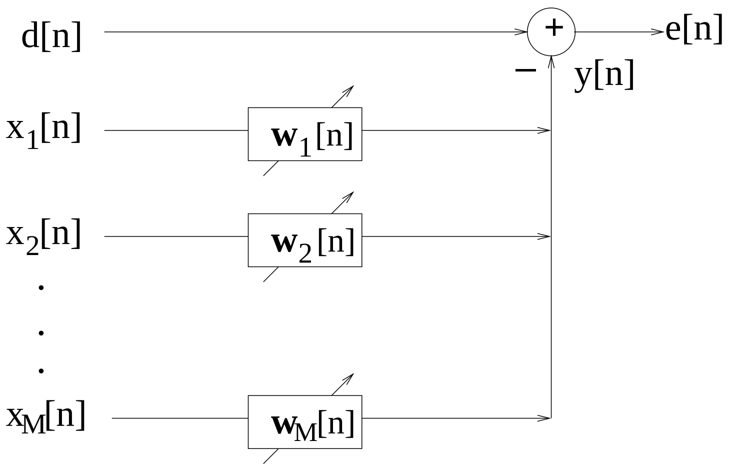

Fig. 5.3: General structure of a multi-channel adaptive filter.

In this part, we focus on the transform-domain method where the QDFT is applied to the input signals and an adaptive algorithm is then applied to the transformed signals by minimizing the error between the output of the adaptive filter and the reference signal. The QLMS algorithm is employed for wind profile prediction to show the improved convergence speed. Moreover, due to correlation of the wind data at different spatial positions, a multi-channel version is used, which will provide further improved performance.

For a standard adaptive filtering structure, as shown before, the error signal $e[n]$ is given by

$$e[n] = d[n] - \mathbf{w}[n]^T\mathbf{x}[n], \tag{5.8}$$

where $d[n]$ is the reference signal, $\mathbf{w}[n]$ denotes the adaptive weight vector with a length of L, $\mathbf{x}[n] = [x[n-1], x[n-2], \cdots, x[n-L]]^T$ is the input data vector, and $\{\cdot\}^T$ denotes the transpose operation.

Instead of the single-channel adaptive filtering structure, in many applications, such as beamforming [22], a multi-channel one has to be employed. Such a structure with M input channels is shown in Fig. 5.3, which also has a particular application in the area of wind profile prediction. In the past, most of the work in linear wind profile prediction has been based on the observation at a single point in space [55, 94]. However, the wind profile at different spatial positions is highly correlated and incorporating information from neighbouring positions in the prediction process will improve performance of the system. This can be realised by a multi-channel linear prediction structure, where the reference signal in Fig. 5.3 will be a properly delayed version of one input signal, with the delay determined by the required prediction step. For the M-channel adaptive filtering structure with L filter taps attached to each channel, the output $y[n]$ is given by

$$y[n] = \mathbf{w}^T[n]\mathbf{x}[n] \ , \tag{5.9}$$

where the $ML \times 1$ coefficients weight vector $\mathbf{w}[n]$ are defined as

$$\mathbf{w}[n] = \left[\mathbf{w}_1{}^T[n] \ \ \mathbf{w}_2{}^T[n] \ \ ... \ \ \mathbf{w}_M{}^T[n]\right]^T \tag{5.10}$$

with

$$\mathbf{w}_m[n] = \left[w_{m,1}[n] \ \ w_{m,2}[n] \ \ ... \ \ w_{m,L}[n]\right]^T \tag{5.11}$$

and the $ML \times 1$ input data vector $\mathbf{x}[n]$ is defined as

$$\mathbf{x}[n] = \left[\mathbf{x}_1{}^T[n] \ \ \mathbf{x}_2{}^T[n] \ \ ... \ \ \mathbf{x}_M{}^T[n]\right]^T \tag{5.12}$$

with

$$\mathbf{x}_m[n] = \left[x_m[n-1] \ \ x_m[n-2] \ \ ... \ \ x_m[n-L]\right]^T . \tag{5.13}$$

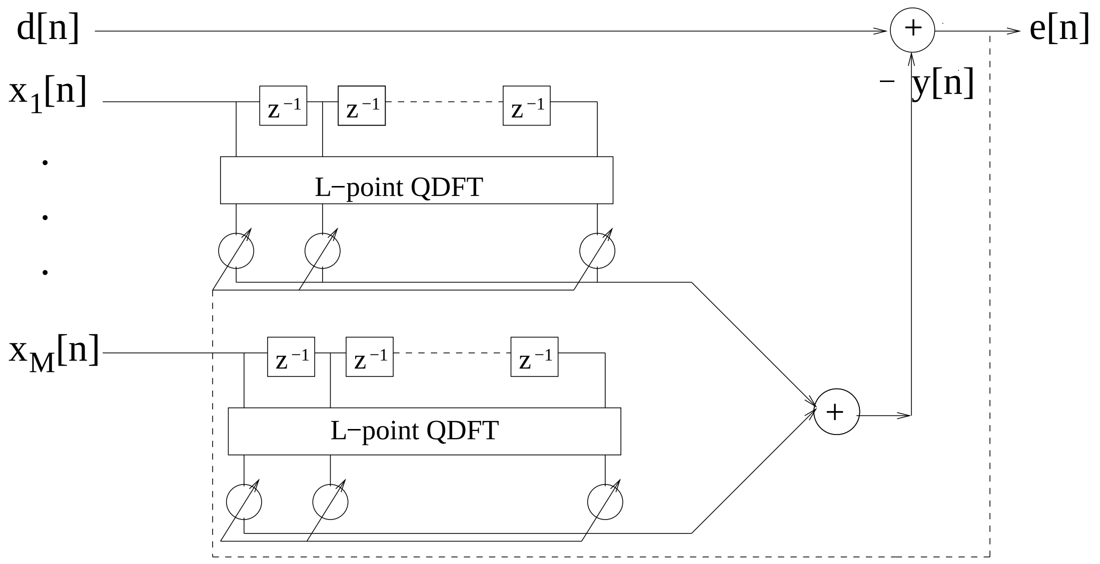

Fig. 5.4: General structure of a multi-channel frequency-domain quaternion-valued adaptive filter.

The update equation for the M-channel case can be derived in a straightforward way from the single-channel one.

There are different structures for tran sform-domain implementation of traditional adaptive filtering algorithms [90], associated with varied advantages and disadvantages. In this work, similar to [95], we apply an L-point QDFT to each of the input signals and the QDFT outputs are then combined together by a weight vector to give the output signal $y[n]$, which is then compared with the reference signal $d[n]$ directly to generate the error signal $e[n]$. The resultant structure is shown in Fig. 5.4.

For the QDFT [26], there are two different types, which are left-side and right-side, respectively, due to the non-commutativity property of a quaternion [50]. In our simulations, we have tried both and they give very similar results. So here we only focus on the right-side QDFT. As a quaternion has three imaginary components, it is essential to specify the transform axis. With the i transform axis, the QDFT of $\mathbf{x}_m$ can be expressed as

$$X_{m,k} = \sum_{l=1}^{L} x_m[n-l]e^{-i2\pi kl/L}, \tag{5.14}$$

where $k = 1, 2, \cdots, L$. All of the L components $X_{m,k}$ form the vector $\mathbf{X}_m = \begin{bmatrix} X_{m,1} & ... & X_{m,L} \end{bmatrix}^T$ and further we have the new data vector in the frequency domain

$$\mathbf{X}[n] = [\mathbf{X}_1{}^T[n] \;\; \mathbf{X}_2{}^T[n] \;\; ... \;\; \mathbf{X}_M{}^T[n]]^T. \tag{5.15}$$

The inverse QDFT (IQDFT) of $\mathbf{X}_m$ is [51]

$$x_m[n-l] = \frac{1}{L}\sum_{k=1}^{L} X_{m,k}e^{i2\pi kl/L}. \tag{5.16}$$

With a simplified notation, the output of the L-point QDFT is

$$\mathbf{X}_m[n] = \text{QDFT}[\mathbf{x}_m[n]] \tag{5.17}$$

and its inverse form is

$$\mathbf{x}_m[n] = \text{IQDFT}[\mathbf{X}_m[n]]. \tag{5.18}$$

Similar to the time-domain QLMS algorithm, the frequency-domain multi-channel QLMS algorithm can be expressed as

$$e[n] = d[n] - \mathbf{w}_f{}^T[n]\mathbf{X}[n] \tag{5.19}$$

$$\mathbf{w}_f[n+1] = \mathbf{w}_f[n] + \frac{1}{2}\mu_7(e[n]\mathbf{X}_f{}^*[n]) \;. \tag{5.20}$$

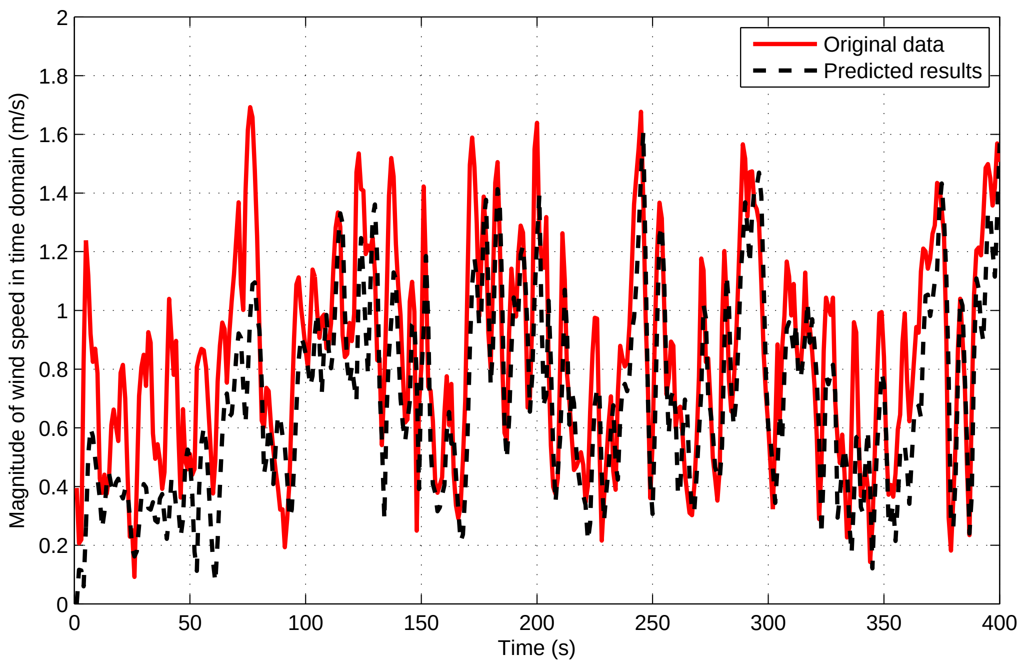

Fig. 5.5: Prediction results using the QLMS algorithm.

We will see in our simulations part that the proposed trasform-domain method can increase the convergence speed significantly due to the decorrelation effect of the transform and therefore in a dynamic environment, provide much improved tracking capability.

5.3 Simulation Results

5.3.1 Results based on CFD generated data

In this part, after performing the correlation analysis, both the QLMS and the AQLMS algorithms are applied to the wind data generated by the CFD software with a sampling frequency of 1 Hz. The parameters are as follows. The step size is $\mu = 2.5 \times 10^{-4}$ and the adaptive filter length is $L = 16$. The prediction step

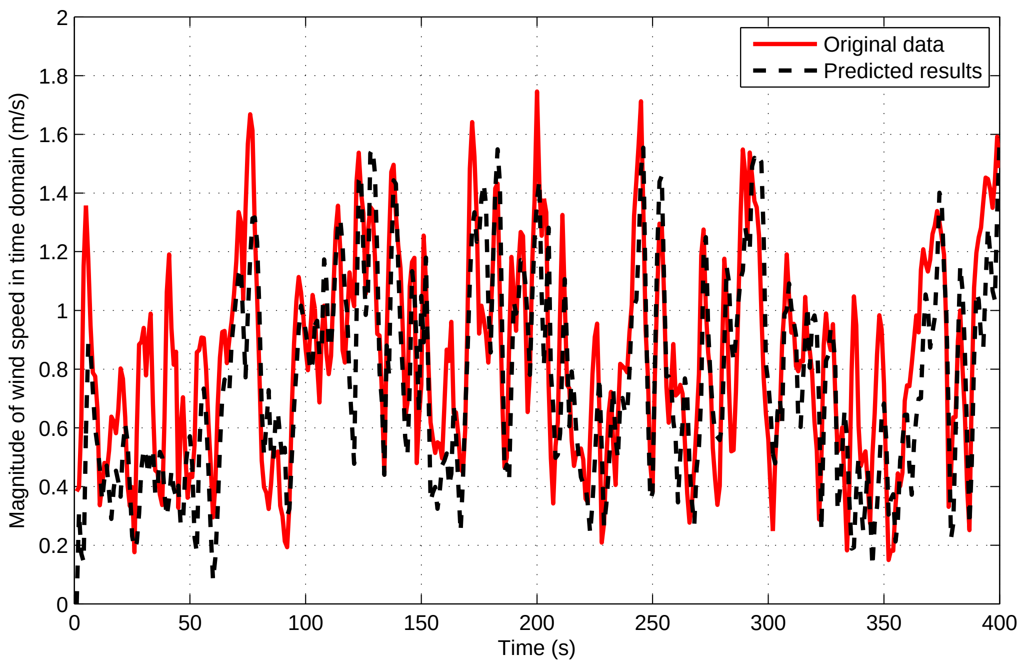

Fig. 5.6: Prediction results using the AQLMS algorithm.

is 2. The adaptive weight vector is initialized as an all-zero vector. Fig. 5.5 and Fig. 5.6 show the results for the QLMS and AQLMS algorithms, respectively. As we can see from the results, both algorithms can track the change of the wind speed signal effectively.

5.3.2 Results based on real data

In this section, simulations will be provided to show the performance of the proposed frequency-domain method based on the 3-D wind data obtained from a Google website [96]. The data was recorded by anemometers at Tracy, CA, US with a 7.6 Hz sampling rate on 11th June, 2011. Two sets of simulations will be performed, one for the single channel structure and one for the multi-channel structure.

Scenario One: Single-channel prediction

In the first set of simulations, we use the wind data recorded at one single location and perform the prediction using past values of the data at the same location. The parameters are as follows. The step size is $\mu = 10^{-6}$ and the adaptive filter length is $L = 8$. The prediction step is 1. The adaptive weight vector is initialized as an all-zero vector.

Fig. 5.7 shows the result for the time-domain QLMS algorithm, with Fig. 5.8 for the frequency-domain QLMS algorithm. Note that the x-axis is the time interval from the start of iterations, and the y-axis is the magnitude of wind speed. Comparing these two results, we can see that the frequency-domain algorithm outperforms the time-domain algorithm, since it can track the original signal more quickly. To show this more clearly, the learning curves of the error signal for the two methods are provided in Fig. 5.9.

Scenario Two: Multi-channel prediction

As to the multi-channel predictor in our second set of simulations, it has $M = 5$ channels and the length of the adaptive filter attached to each channel is $L = 8$. The step size is $\mu = 5 \times 10^{-6}$ and the prediction step is $P = 1$. Similarly, the initial adaptive weight vector is set to be of all zeros.

Compared with Fig. 5.7, the multi-channel result in Fig. 5.10 gives a better prediction performance, due to much more information involved in the prediction process. In Fig. 5.11, it shows the frequency-domain result, with a similar curve to the single-channel case in Fig. 5.8. The learning curves for the predic-

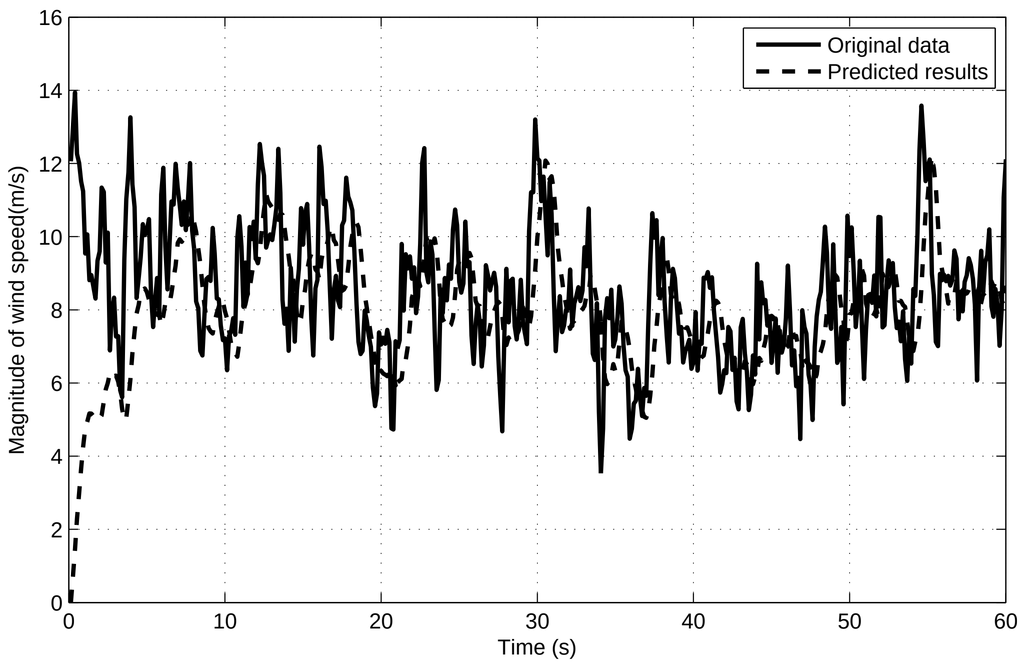

Fig. 5.7: The single-channel time-domain prediction result.

tion error have been shown in Fig. 5.12, where again we can observe that the frequency-domain method has provided a much faster convergence speed.

From the two sets of simulations, we can see that the proposed quaternion-valued multi-channel transform-domain adaptive filter is very promising in applications requiring a fast tracking capability, and clearly a better choice for the specific area of wind profile prediction.

For the adaptive filter length chosen in this section, both of the above scenarios are set to $L = 8$. The following figures are the comparison results between different filter lengths (L=16, 32, 64) in time-domain and transform-domain, respectively. From the result we can see that the convergence speed is faster for the relatively larger filter length with similar steady state error, however, there is a trade-off between filter length and error. With the increasing L, the learning

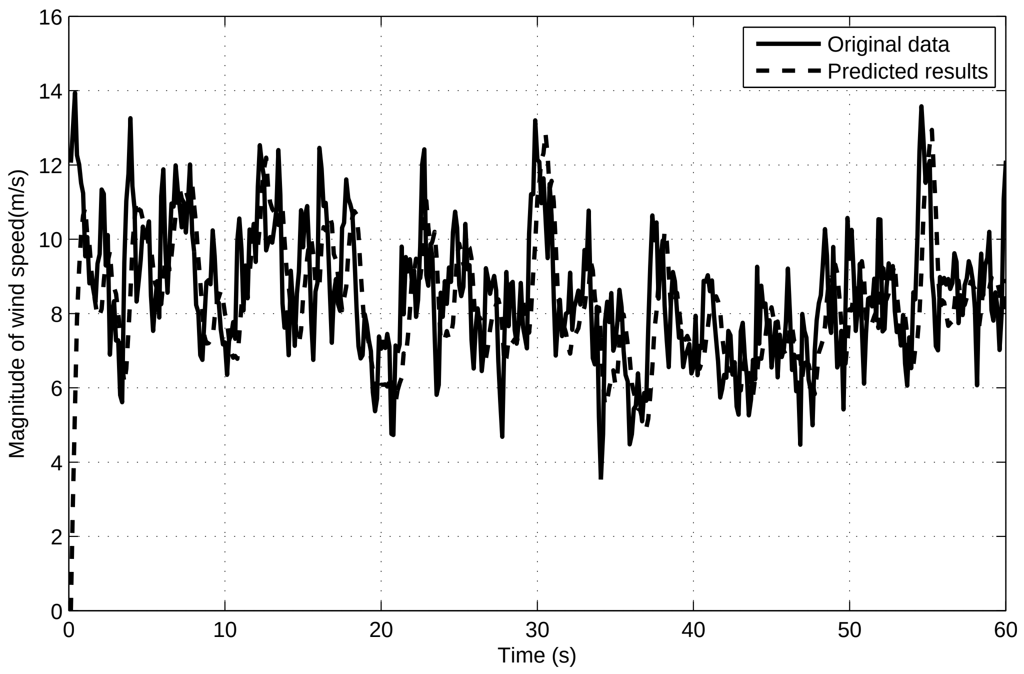

Fig. 5.8: The single-channel transform-domain prediction result.

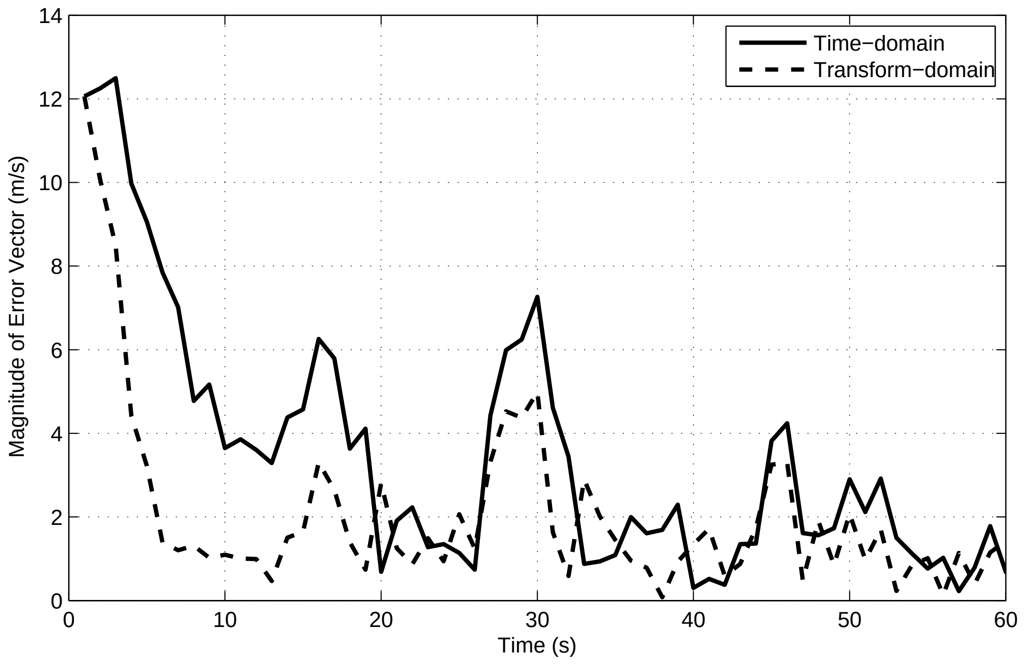

Fig. 5.9: The single-channel learning curve for the magnitude of the error signal.

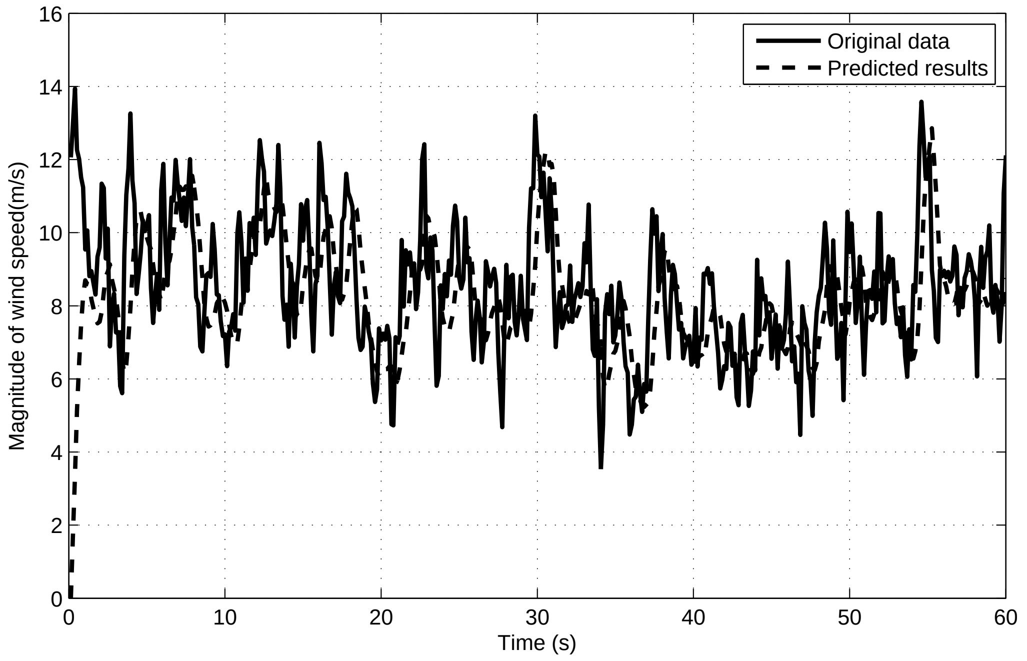

Fig. 5.10: The multi-channel time-domain prediction result.

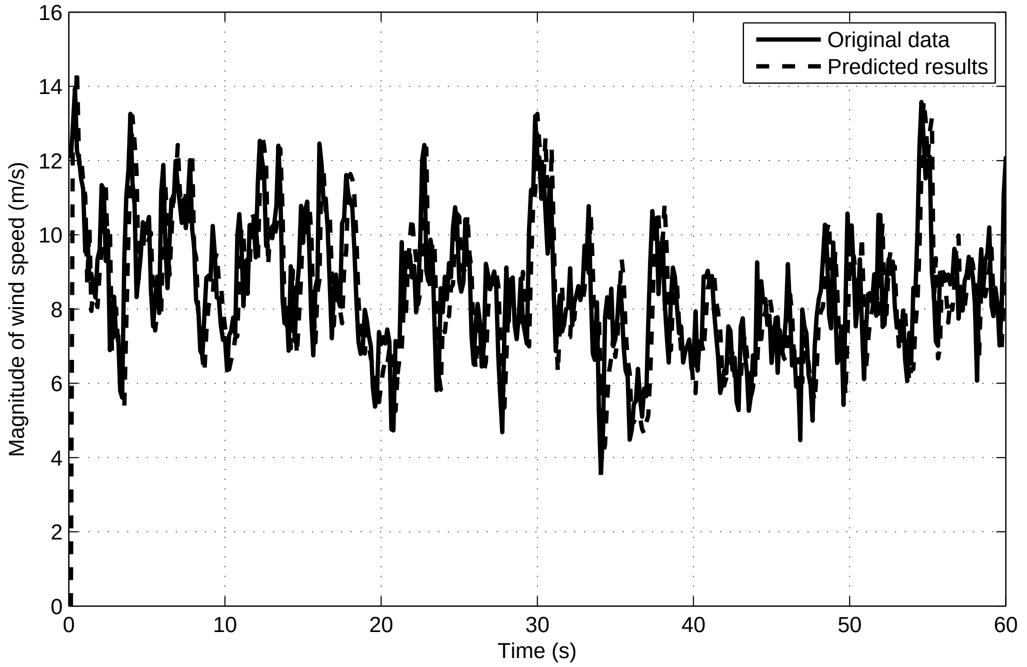

Fig. 5.11: The multi-channel transform-domain prediction result.

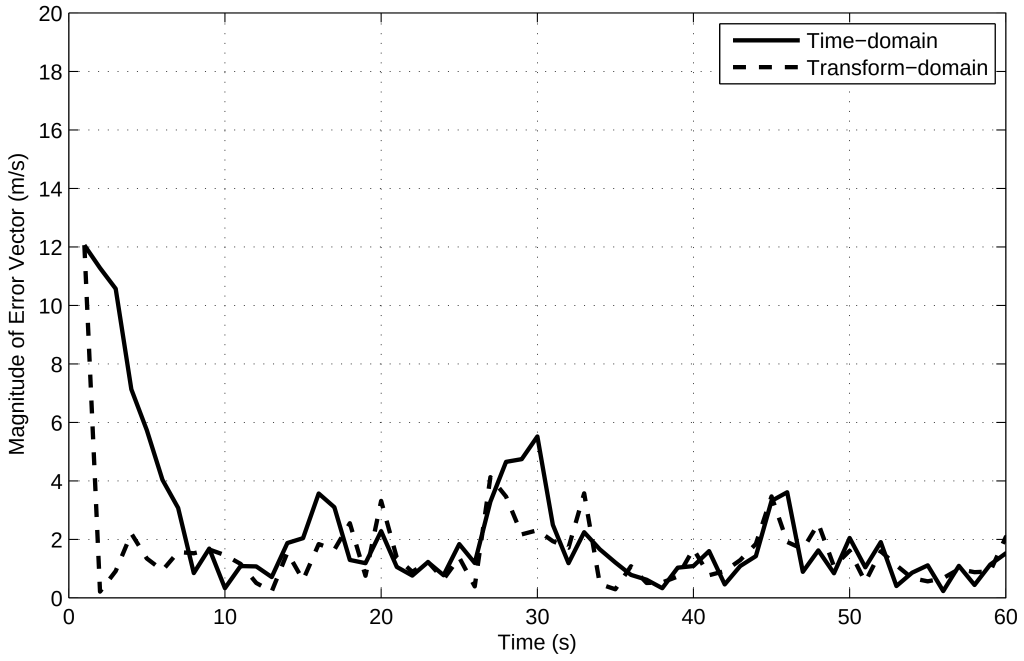

Fig. 5.12: The multi-channel learning curve for the magnitude of the error signal.

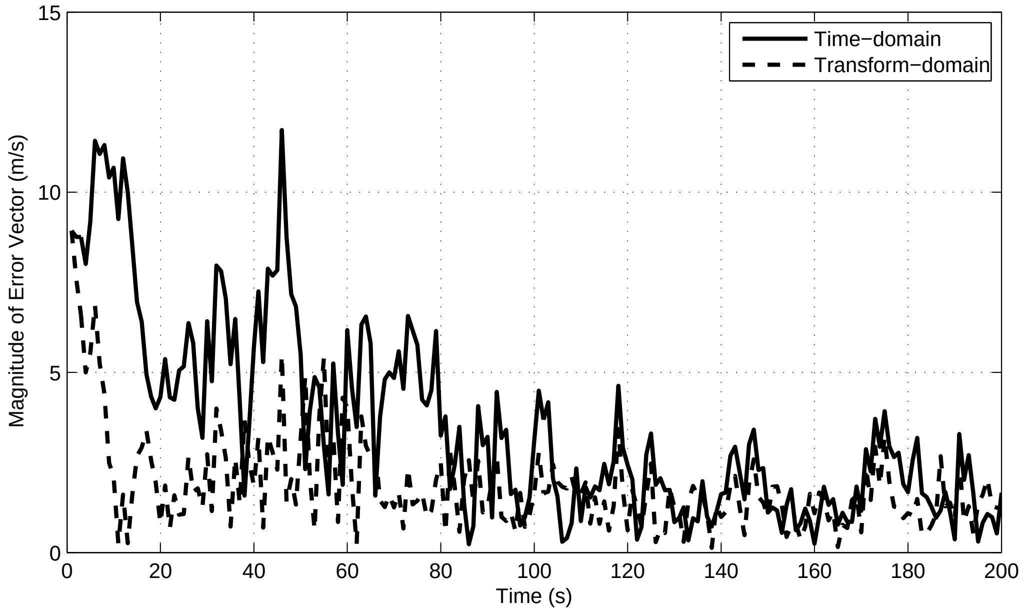

Fig. 5.13: The learning curve for the magnitude of the error signal with L=16.

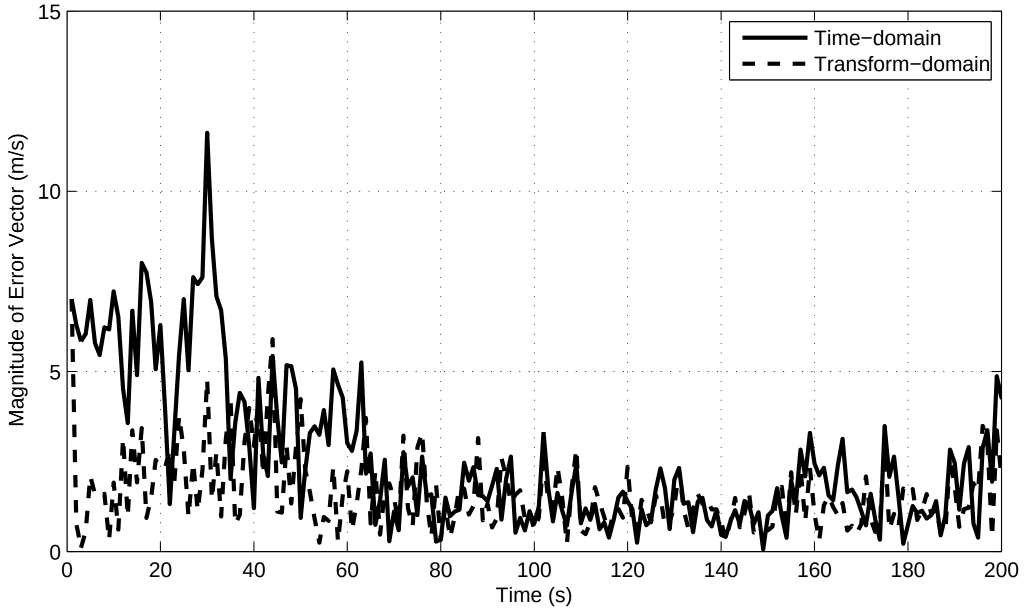

Fig. 5.14: The learning curve for the magnitude of the error signal with L=32.

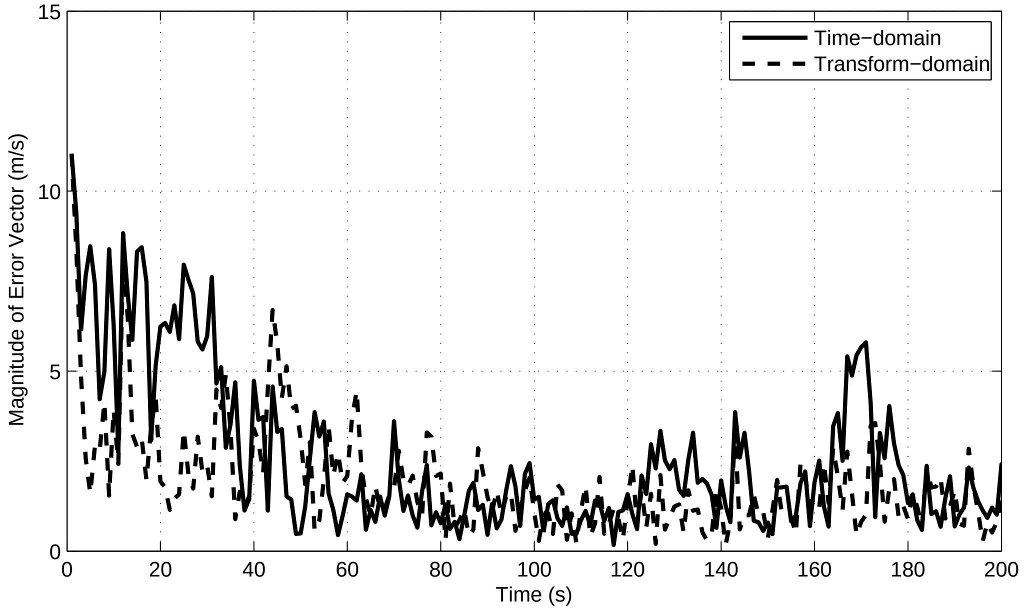

Fig. 5.15: The learning curve for the magnitude of the error signal with L=64.

curve can not converge faster with very large filter length, such as the transform-domain with $L = 64$ in Fig. 5.15 when compared to $L = 32$ in Fig. 5.14. It is obvious that the $L = 32$ shows the better tracking ability than $L = 16$ and $L = 64$.

5.4 Summary

A correlation analysis has been performed using the Wiener solution to show that the signal processing based wind profile prediction method can provide a feasible solution to the underlying problem. With the analysis result, we can find out the appropriate sampling frequency and then use the QLMS/AQLMS algorithm to predict the wind speed effectively. Furthermore, based on the QDFT, we have extended the recently proposed time-domain quaternion-valued adaptive filter into the frequency domain and the single-channel structure to the multi-channel case with a particular application to the area of wind profile prediction. Like the traditional frequency-domain methods, the proposed one can improve the convergence speed of the employed adaptive algorithms, leading to much better tracking capability for applications in a dynamic environment. Simulations based on the QLMS/AQLMS algorithm have been performed using the CFD generated data to show the good prediction result, and also the recorded wind data from multiple measurement sites clearly demonstrate the potential of the proposed method in this particular area.

Chapter 6

Combined Approach to Wind Profile Prediction

As mentioned in last chapter, wind profile prediction is a classical signal prediction problem; on the other hand, wind or atmospheric flow analysis is also a traditional problem in Computational Fluid Dynamics (CFD) in applied mathematics, which employs conservation laws, physical models and numerical methods to make predict. The CFD approach is relatively accurate when compared to other approaches, but there are still some disadvantages. For example, it is time-consuming with high computational complexity, and it contains uncertainties/errors in initial/boundary conditions as well as models. Therefore, the aim of this study is to propose efficient and effective methods for wind profile/atmospheric flow prediction based on synergies between the statistical signal processing approach and the CFD approach.

CFD is a branch of fluid mechanics, which adopts numerical approaches and algorithms to tackle the fluid flow problems. Usually, computers are used to solve the equations that model the motions of liquids and gases with suitable boundary conditions. A large number of fluid dynamics problems are modeled by the Navier-Stokes equations, which will be described in detail in this chapter, together with several CFD techniques.

This chapter is organised as follows. The basic CFD concepts and simulation methods will be reviewed in Section 6.1. Two combined methods (alternating and weighting) based on both the signal processing approach and the CFD approach will be investigated in Section 6.2. A summary of this chapter is given in Section 6.3.

6.1 Computational Fluid Dynamics

6.1.1 The fluid dynamics equations

The Navier-Stokes equations are the basis of fluid dynamics problems. They are essentially the mathematical formulation of the Newton's second law applied to fluid motions. The general expression of the equations is

$$\rho(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}) = -\nabla P + \eta\mu\mathbf{u} \tag{6.1}$$

where **u** is the fluid velocity at a particular spatial location at a given time, P is the pressure and ρ is the fluid density. The left hand side of the equation is the acceleration of the fluid, whilst on the right side are (the gradient of) the forces, including pressure and viscous force. Together with the conservation of mass and suitable boundary conditions, the Navier-Stokes equations can model a large class of fluid motions accurately [97].

6.1.2 Reynolds number

In fluid mechanics, the Reynolds number (Re) is a dimensionless number which is the ratio of the inertial force to the viscous force, and thus measures the relative importance of these two forces with the given flow conditions. The expression for the Reynolds number can be derived from Navier-Stokes equations when all the parameters in the equations are non-dimensionalized properly, which is:

$$Re = \frac{\rho v L}{\mu} = \frac{vL}{\nu} \tag{6.2}$$

where

v is the velocity scale of the fluid (m/s),

L is a characteristic linear dimension of the fluid field,the length scale (m),

μ is the dynamic viscosity of the fluid,

ν is the kinematic viscosity($\nu = \mu/\rho$),

ρ is the density of the fluid.

Therefore, the time scale is $T = L/v$ and the pressure scale is $p = \rho v^2$.

Assuming that the parameters are non-dimensionalized as follows: $\mathbf{u}^* = \mathbf{u}/v$, $\mathbf{x}^* = \mathbf{x}/L$ and $t^* = t/T = tv/L$, the Navier-Stokes equations can be rewritten as

$$\frac{\partial \mathbf{u}^*}{\partial t^*} + (\mathbf{u}^* \cdot \nabla^*)\mathbf{u}^* = -\nabla^* P^* + \frac{1}{Re}\Delta^* \mathbf{u}^* \tag{6.3}$$

where Re is shown in Eq. (6.2). From Eq. (6.3), it is clear that the controlling parameters appear in the form of the Reynolds number. The equation is the same for the same Reynolds number, even if the parameters are different. That is to say, two flow fields will be similar, if their Reynolds numbers are the same.

6.1.3 Turbulence or Laminar

The second term on the left hand side of Eq. (6.1) represents the contribution from the advection of fluid particles to fluid acceleration, and is customarily called the inertial force. The second term on the right hand side represents the viscous force. The ratio of these two forces is the Reynolds number (Re). As it turns out, when Re is large, the flows tend to become unstable and generate a spectrum of high frequency components in the velocity signal. Such a regime of fluid motions is called turbulence. Turbulent flows always occur at high Reynolds numbers [98]. Atmospheric flows, including the wind fields around wind farms, are always turbulence [99]. Due to the presence of the high frequency components, the CFD calculation of the velocity signal in turbulent wind fields becomes very time consuming unless simplifying models are introduced.

6.1.4 Discretisation methods

Finite difference

In fluid mechanics, people have found relatively effective numerical methods to solve the ordinary and also the partial differential equations. They start with some forms of discretisation and one of these methods is the finite difference method. The detailed calculation process can be described as follows.

We use the following simple ordinary differential equation to illustrate the method

$$\frac{d^2y}{dx^2} = -\lambda y \ . \tag{6.4}$$

We assume that the boundary values $y(0) = y_a$, $y(1) = y_b$ are known, and suppose we want to find the solutions at $(x_1, x_2, \cdots, x_N)$. Then, we need the appropriate equations for the unknown $(y_1, y_2, \cdots, y_N)$, where y_n is the solution at x_n. In terms of these unknown values, it is quite simple to derive the approximate form of the ordinary differential equations.

One of the approximate forms for $\frac{d^2y}{dx^2}$ is [100]:

$$\left(\frac{d^2y}{dx^2}\right)_{x_n} = \frac{y_{n-1} + y_{n+1} - 2y_n}{h^2} \ . \tag{6.5}$$

At the point x_n, the relation between y_n, y_{n-1} and y_{n+1} follows from Eq. (6.4):

$$\frac{y_{n-1} + y_{n+1} - 2y_n}{h^2} = -\lambda y_n$$

$$y_{n-1} + y_{n+1} - 2y_n = -\lambda h^2 y_n \ . \tag{6.6}$$

As x_1 and x_N are set to be 0 and 1, respectively, the unknown values for y_n can be calculated by means of iteration. More specifically, when using the former equation, at point x_2, the expression is

$$y_3 + y_1 - 2y_2 = -\lambda h^2 y_2 \ . \tag{6.7}$$

For point x_{N-1}, the description of the equation will be

$$y_N + y_{N-2} - 2y_{N-1} = -\lambda h^2 y_{N-1} \ . \tag{6.8}$$

Due to the given condition $y_1 = y_a$ and $y_N = y_b$, in total, we have $N - 2$ unknown variables with $N - 2$ algebraic equations. Therefore, those variables can be obtained easily. As mentioned before, the values $y_1, y_2, \cdots, y_N$ provide an approximation to the solution y_x of the ordinary differential equation. The approximation can be improved by increasing the number of grid points N.

Spectral method

Another method is spectral method, which will be used in our programme. The Fourier series for a function $u(x)$ can be expressed as follows with $0 \leq x \leq 2\pi$ and $k_n = n\pi$

$$u(x) = \sum_{n=-\infty}^{+\infty} \hat{u}_n e^{ik_n x} \tag{6.9}$$

where $\hat{u}_n = \frac{1}{2\pi}\int_0^{2\pi} u(x)e^{-ik_n x}\,dx$ is the Fourier coefficient. When we try to find numerical solutions, the series has to be truncated. Thus, $u(x)$ will be approximated by the following equations with a sufficiently large N

$$u(x) = \sum_{n=-N}^{N} \hat{u}_n e^{ik_n x} \; . \tag{6.10}$$

In the spectral method, we aim at solving $\hat{u}_n$. When $\hat{u}_n$ is known, $u(x)$ is given by Eq. (6.10). Therefore, we need to derive the equation for $\hat{u}_n$. We use the Burger's equation to illustrate the idea and the Burger's equation is given by

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} \; . \tag{6.11}$$

The Burger's equation is a 1-D model of the Navier-Stokes equations, but it retains the most important nonlinear term. Therefore, it is an essential model to study the nonlinear effects in the Navier-Stokes equations.

Now we derive the equation for $\hat{u}_n$ based on the Burger's equation:

$$\begin{aligned}\int_0^{2\pi} e^{-ik_n x}\left(\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}\right)dx &= \int_0^{2\pi} e^{-ik_n x}\frac{\partial u}{\partial t}\,dx+\int_0^{2\pi} e^{-ik_n x}u\frac{\partial u}{\partial x}\,dx \\ &= \frac{\partial \hat{u}_n}{\partial t}+(\widehat{u\frac{\partial u}{\partial x}})_n \end{aligned} \tag{6.12}$$

In practice, $u\frac{\partial u}{\partial x}$ is evaluated with a pseudo-spectral method [101]. This viscous diffusion term becomes:

$$\begin{aligned}&\int_0^{2\pi} e^{-ik_n x}\left(\nu\frac{\partial^2 u}{\partial x^2}\right)dx \\ =\ & \nu(e^{-ik_n x}\left(\frac{\partial u}{\partial x}\right)\Big|_0^{2\pi}-\int_0^{2\pi} ue^{-ik_n x}(-ik_n x)(\nu\frac{\partial u}{\partial x})\,dx \\ =\ & \int_0^{2\pi}\nu e^{-ik_n x}(ik_n x)(\frac{\partial u}{\partial x})\,dx \\ =\ & \nu(ik_n)^2\hat{u}_n \\ =\ & -\nu k_n^2\hat{u}_n\ . \end{aligned} \tag{6.13}$$

The equation in the Fourier series now becomes

$$\frac{\partial \hat{u}_n}{\partial t}+(\widehat{u\frac{\partial u}{\partial x}})_n=-\nu k_n^2\hat{u}_n, \tag{6.14}$$

in other words, a set of ordinary differential equations, which are then solved by finite difference approximation in time.

6.1.5 Simulations methods and models for turbulence

Direct numerical simulation

Direct numerical simulation (DNS) solves the Navier-Stokes equations using a pseudo-spectral method directly without any turbulence models [102]. The computational cost for it can be very high even if Re is moderate. Therefore, this method is not yet applicable to practical situations [97], such as, the atmospheric flows we will deal with. However, the advantage of this method is that it is simple as well as accurate with complete information, and it is useful in the development of turbulence models for practical applications. Therefore, as a first step, we use DNS to generate the reference velocity signals to compare with the results produced by other models.

Large eddy simulation (LES) and Subgrid-scale model (SGS)

Large eddy simulation (LES) is a mathematical model for turbulence proposed by Joseph Smagorinsky to simulate atmospheric air currents [103], and first explored by Deardorff in 1970 [104]. In LES we eliminate the small length scales of the solution and simulate the large scales only, which reduces the calculation cost. However, because of the nonlinear nature of the Navier-Stokes equations, the small scales are coupled to the large scales. The effects have to be modeled; otherwise, the solution will diverge. The model is called subgrid-scale model.

To separate large scales from small scales, the idea of low pass filtering (LPF) is applied. As a result, the input $u(x)$ becomes output $\tilde{u}(x)$ through the LPF with system function $G(x)$. More specifically, the filtering process is expressed as follows:

$$\tilde{u}(x) = G(x) * u(x) = \int_{-\infty}^{+\infty} G(y-x)u(y)\,dy \tag{6.15}$$

where $\tilde{u}(x)$ is the filtered velocity, which is what we solve in LES. We need the equation for $\tilde{u(x)}$. Again using the Burgers equation as an example, we apply the filter to the Burgers equation:

$$\begin{aligned}\int_{-\infty}^{+\infty} G(y-x)(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial y})\,dy &= \int_{-\infty}^{+\infty} G(y-x)\nu\frac{\partial^2 u}{\partial x^2}\,dy \\ &= \frac{\partial}{\partial t}\big(\int_{-\infty}^{+\infty} G(y-x)u\,dy\big) + \widetilde{u\frac{\partial u}{\partial x}} \\ &= \nu\frac{\partial^2 \tilde{u}}{\partial x^2} \end{aligned} \tag{6.16}$$

with

$$\widetilde{u\frac{\partial u}{\partial x}} = \int_{-\infty}^{+\infty} G(y-x)u\frac{\partial u}{\partial y}\,dy. \tag{6.17}$$

We obtain the resultant 'filtered' Burgers equation:

$$\frac{\partial \tilde{u}}{\partial t} + \widetilde{u\frac{\partial u}{\partial x}} = \nu\frac{\partial^2 \tilde{u}}{\partial x^2} \tag{6.18}$$

From

$$u\frac{\partial u}{\partial x} = \frac{\partial}{\partial x}(\frac{u^2}{2}), \tag{6.19}$$

we get

$$\widetilde{u\frac{\partial u}{\partial x}} = \frac{\partial}{\partial x}(\widetilde{\frac{u^2}{2}}). \tag{6.20}$$

Therefore, equation (6.18) becomes:

$$\frac{\partial \tilde{u}}{\partial t} + \frac{1}{2}\frac{\partial}{\partial x}\tilde{u^2} = \nu\frac{\partial^2 \tilde{u}}{\partial x^2} \tag{6.21}$$

However, as $\tilde{u^2}$ and $\tilde{u}^2$ are different, the equation is not closed. In LES, we write the equation in the following form:

$$\begin{aligned}\frac{\partial \tilde{u}}{\partial t} + \tilde{u}\frac{\partial \tilde{u}}{\partial x} &= \nu\frac{\partial^2 \tilde{u}}{\partial x^2} + \tilde{u}\frac{\partial \tilde{u}}{\partial x} - \frac{1}{2}\frac{\partial}{\partial x}\tilde{u^2} \\ &= \nu\frac{\partial^2 \tilde{u}}{\partial x^2} - \frac{1}{2}\frac{\partial}{\partial x}(\tilde{u^2} - \tilde{u}^2)\end{aligned} \tag{6.22}$$

where the term $\widetilde{u^2} - \tilde{u}^2$ is called the subgrid-scale stress. Since it is not closed, we need to model it in terms of $\tilde{u}$. The model is what we called subgrid-scale model. For Navier-Stokes equations, we will have a subgrid-scale stress tensor, which is denoted as τ_{ij}:

$$\tau_{ij} = \widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j \tag{6.23}$$

In our study, we will also use LES to model the wind fields [97].

6.2 The Combined Approach

In the combined method, the signal processing part employs the QLMS algorithm, while for the CFD part, LES based on the Smagorinsky SGS model will be employed. There are different ways of combining them together to obtain a more effective and efficient method for wind profile prediction. In our current study, we mainly focus on the issue of efficiency, i.e. we aim to develop a method which can achieve a similar level of accuracy as the CFD approach but with a lower complexity. Certainly, it is possible to increase the complexity of the new method a little (but still lower than the original CFD approach) and achieve a more accurate result and in this case the new method could be more efficient and at the same time more effective as well.

To combine the QLMS algorithm and LES together, two approaches are adopted in our study: one is to combine the results of QLMS prediction and LES by an appropriate weighting function and the other is to alternate their operations in succession. Here we will mainly focus on the alternating method and show that its running time is much shorter than the CFD method while still maintaining a comparable good prediction result. For the weighting method, we have done some preliminary analysis, which does give some good results; nevertheless, more analysis and research are needed in the future.

6.2.1 Alternating

The alternating process is described as follows. First, the QLMS algorithm is used to obtain the predicted wind velocity, and then we use the prediction result as the initial conditions for the LES method for calculating the next stage of the wind velocity. In the next round, the same process is repeated. The process flow chart is shown below in Fig. 6.1.

As the LES approach is more accurate over the time scale of large scale flows but time consuming with a high computational cost, and the QLMS algorithm has a very low complexity, an alternating combination of these two will produce a method with a much lower complexity.

When comparing the accuracy of the new method with the CFD approach (the LES method in this context), the CFD data generated by DNS is used as a reference. The DNS simulation is based on a denser grid than the LES: $64 \times 64 \times 64$ grid points for DNS and $32 \times 32 \times 32$ grids for LES. The code is written in FORTRAN 90. Running the code, we can then generate a time series of three-dimensional turbulent wind velocity fields.

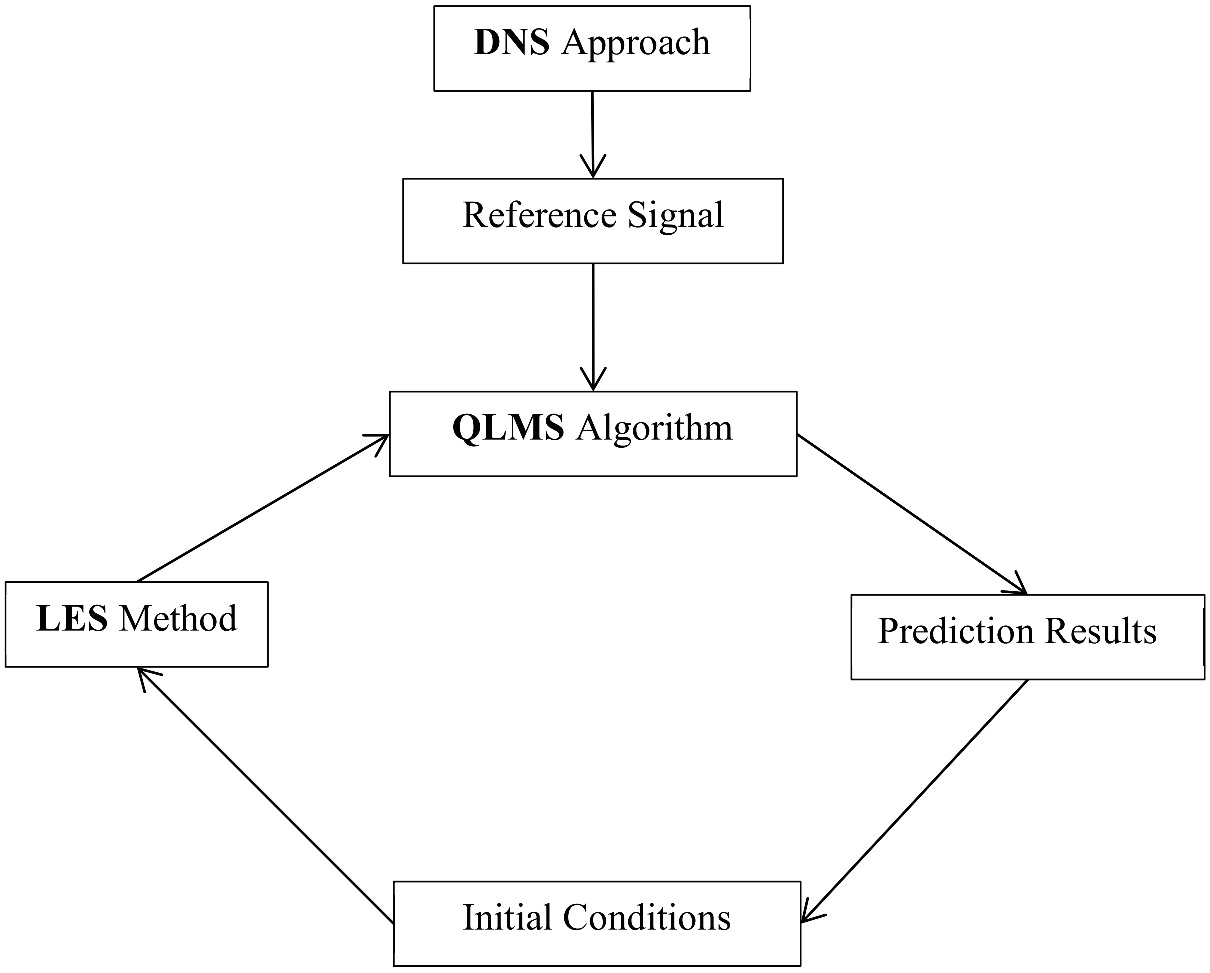

Fig. 6.1: The alternating progress.

Simulation

Simulation results are provided in this section to show the performance of the proposed alternating method. Based on an analysis of the correlation generated by the CFD data, we have chosen the sampling frequency as 1/6 Hz for this data set, i.e. the sampling interval between adjacent samples is 6 seconds. The other parameters are: the length of the FIR filter is $L = 16$ and the step size is $\mu = 1 \times 10^{-5}$.

The results are shown in Table 6.1, where the first column is the prediction advance value P, e_1 the normalised error between the LES method and the DNS method, and e_2 is the normalised error between the combined method and the DNS method. We can see that in terms of the normalised prediction error, the two methods have a very similar performance. As to the computational complexity, we show the running time of the two methods in Table 6.2, where t_1 and t_2 are for the LES method and the alternating method, respectively. We can see that the running time for the LES method is nearly doubled when compared to the alternating method as the prediction advance step increases, highlighting the clear advantage of the proposed combined alternating method.

6.2.2 Weighting

For the weighting method, we use the QLMS algorithm based on the DNS wind data to obtain the prediction result. Meanwhile, the LES method will be employed to predict the wind velocity as well. As the LES approach is accurate

Table 6.1: Normalised prediction error for the proposed alternating method with prediction advance value P.

P	e_1	e_2
1	0.0538	0.0854
2	0.0969	0.0762
3	0.0889	0.0811
4	0.1115	0.0842
5	0.0880	0.0759
6	0.0861	0.0924
7	0.1042	0.1028
8	0.1013	0.0973
9	0.1022	0.0871
10	0.0960	0.0820

Table 6.2: Running time with prediction advance value P (seconds).

P	t_1	t_2
1	260.9	214.1
2	484.6	380.4
3	832.2	533.2
4	1432.5	657.4
5	1813.6	787
6	1951.4	1015.9
7	2338.9	1253.2
8	2465	1177.1
9	2489.3	1464.7
10	3134.3	1646.4

Table 6.3: Normalised prediction error in (dB) by data sequence with sampling frequency f_s (Hz) and prediction time P_t (hours) (Part I).

	P_t=0.5 hours	P_t=1 hours	P_t=1.5 hours	P_t=2 hours
f_s=1Hz	-24.7813	-24.5924	-24.3497	-24.6158
$f_s=\frac{1}{3}$ Hz	-24.6510	-24.7694	-24.5924	-24.5341
$f_s=\frac{1}{6}$ Hz	-24.3383	-24.2928	-24.2702	-24.3497
$f_s=\frac{1}{7}$ Hz	-24.4415	-24.5690	-24.5341	-24.0795
$f_s=\frac{1}{8}$ Hz	-24.0241	-23.5283	-23.9580	-23.9470
$f_s=\frac{1}{9}$ Hz	-23.8380	-24.0573	-23.9032	-24.0795
$f_s=\frac{1}{12}$ Hz	-23.7623	-23.4968	-24.1800	-23.2483
$f_s=\frac{1}{18}$ Hz	-23.0259	-22.4810	-21.3031	-21.6980

but time consuming with high computational cost, and the QLMS algorithm is relatively simple in simulation, and the different weighting between them can be applied to reduce the complexity as well as maintaining a good prediction result.

Firstly, we will show the normalised error of the wind velocity with 8 different sets. The first sampling frequency f_s is set as 1Hz. Then the rest sets are at 1/3, 1/6, 1/7, 1/8, 1/9, 1/12, 1/18 Hz, respectively.

With the above analysis, we can obtain tables with sampling frequency along column direction and prediction time in the row direction, and the values in the tables are the resultant prediction error normalised by the corresponding data sequence, which are shown in Table 6.3 and Table 6.4. Moreover, in Table 6.5 and Table 6.6, the rows and columns are the same, and the values are the resultant prediction error normalised by the power of the corresponding data sequence.

Table 6.4: Normalised prediction error in (dB) by data sequence with sampling frequency f_s (Hz) and prediction time P_t (hours) (Part II).

	P_t=2.5 hours	P_t=3 hours	P_t=3.5 hours	P_t=4 hours
f_s=1 Hz	-24.6158	-24.5225	-24.5690	-24.6510
f_s=$\frac{1}{3}$ Hz	-24.5574	-24.7337	-24.6393	-24.7694
f_s=$\frac{1}{6}$ Hz	-24.0351	-24.3955	-24.4993	-24.2475
f_s=$\frac{1}{7}$ Hz	-23.9580	-24.0795	-24.0351	-24.3156
f_s=$\frac{1}{8}$ Hz	-23.9800	-23.6872	-23.9360	-24.3726
f_s=$\frac{1}{9}$ Hz	-23.5073	-23.5916	-23.7194	-23.7947
f_s=$\frac{1}{12}$ Hz	-22.8573	-23.9142	-23.5810	-23.6127
f_s=$\frac{1}{18}$ Hz	-22.1824	-22.6144	-22.2285	-22.1457

Table 6.5: Normalised prediction error in (dB) by the power of data sequence with sampling frequency f_s (Hz) and prediction time P_t (hours) (Part I).

	P_t=0.5 hours	P_t=1 hours	P_t=1.5 hours	P_t=2 hours
f_s=1 Hz	-12.9811	-12.9437	-12.9048	-12.9538
f_s=$\frac{1}{3}$ Hz	-11.8369	-11.8611	-11.8344	-11.8262
f_s=$\frac{1}{6}$ Hz	-11.0367	-11.0332	-11.0271	-11.0483
f_s=$\frac{1}{7}$ Hz	-10.8795	-10.9174	-10.9219	-10.8347
f_s=$\frac{1}{8}$ Hz	-10.6647	-10.5723	-10.6578	-10.6505
f_s=$\frac{1}{9}$ Hz	-10.4722	-10.5315	-10.4955	-10.5276
f_s=$\frac{1}{12}$ Hz	-10.1484	-10.1106	-10.2347	-10.0493
f_s=$\frac{1}{18}$ Hz	-9.4960	-9.3864	-9.1627	-9.2524

Table 6.6: Normalised prediction error in (dB) by the power of data sequence with sampling frequency f_s (Hz) and prediction time P_t (hours) (Part II)

	P_t=2.5 hours	P_t=3 hours	P_t=3.5 hours	P_t=4 hours
f_s=1 Hz	-12.9591	-12.9364	-12.9458	-12.9570
f_s=$\frac{1}{3}$ Hz	-11.8230	-11.8531	-11.8407	-11.8669
f_s=$\frac{1}{6}$ Hz	-10.9924	-11.0531	-11.0763	-11.0238
f_s=$\frac{1}{7}$ Hz	-10.8056	-10.8171	-10.8076	-10.8623
f_s=$\frac{1}{8}$ Hz	-10.6592	-10.6079	-10.6369	-10.7226
f_s=$\frac{1}{9}$ Hz	-10.4500	-10.4511	-10.4768	-10.4850
f_s=$\frac{1}{12}$ Hz	-9.9828	-10.1862	-10.1162	-10.1342
f_s=$\frac{1}{18}$ Hz	-9.3347	-9.4004	-9.3310	-9.3228

From the table shown in 6.3 and 6.4, we can see that the correlation is smaller when sampling frequency is decreased, as expected. Therefore, the normalised error has been increased as the value of prediction step P increases or the sampling frequency f_s decreases. Therefore, we can make prediction further in advance with smaller error when the correlation value is large, and the acceptable prediction step value depends on the correlation of these samples. On the other hand, we can estimate the prediction advance time through analysing the correlation distribution of the generated data.

Simulations

In terms of the given data set, we can choose the 1/9 Hz to be the sampling frequency of the following predictions. The histograms of error signal for both methods will be shown below. The parameters are: the length of the FIR filter

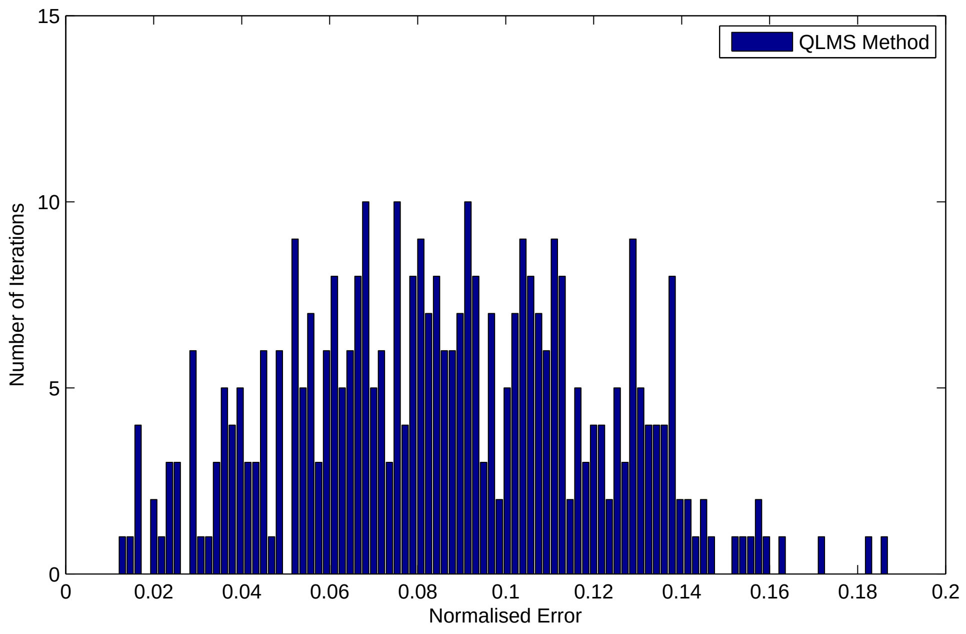

Fig. 6.2: Histograms for normalised error with the QLMS algorithm.

is $L = 16$ and step size is $\mu = 1 \times 10^{-5}$. The prediction advance step is $N = 1.6 \times 10^3$, which accounts for 4 hours prediction advance time.

From the histogram bars, we can see that the distributions are rather different. Therefore, we have to try different weighting to find out the relatively small error signals. Then, in the following step, different constant weighting between the QLMS algorithm and the LES method will be used, such as, 0.3, 0.5, 0.7. Then the DFT will be applied to the remaining normalised error signal to see their frequency components.

These three different weightings have been employed and their normalised prediction errors are shown in Figs. 6.4, 6.5 and 6.6. Moreover, their corresponding spectrum of the error signal is shown in Figs. 6.7, 6.8 and 6.9. From these figures, we can see that the error is smaller when the weighting for QLMS

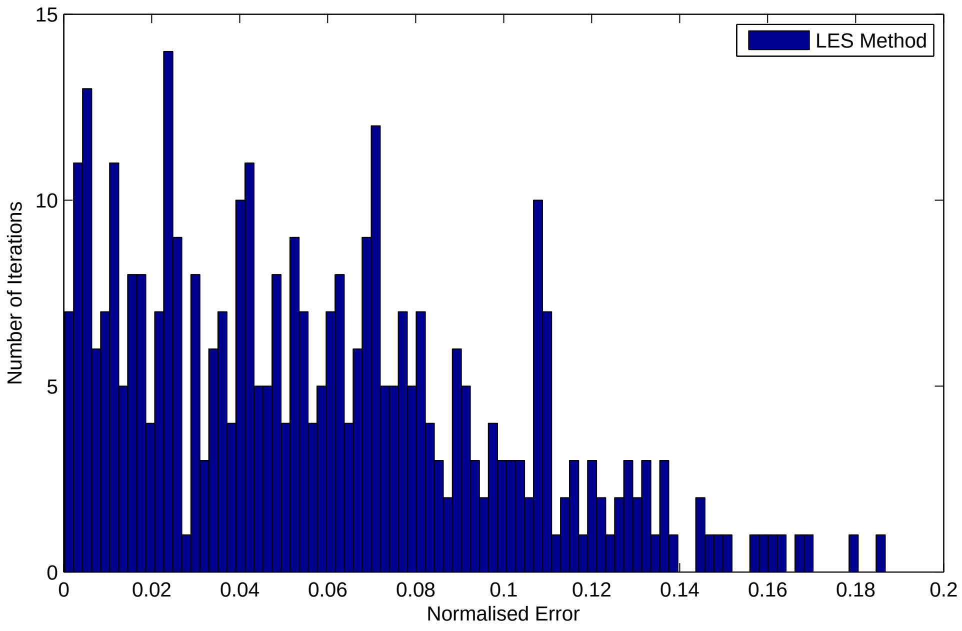

Fig. 6.3: Histograms for normalised error with the LES method.

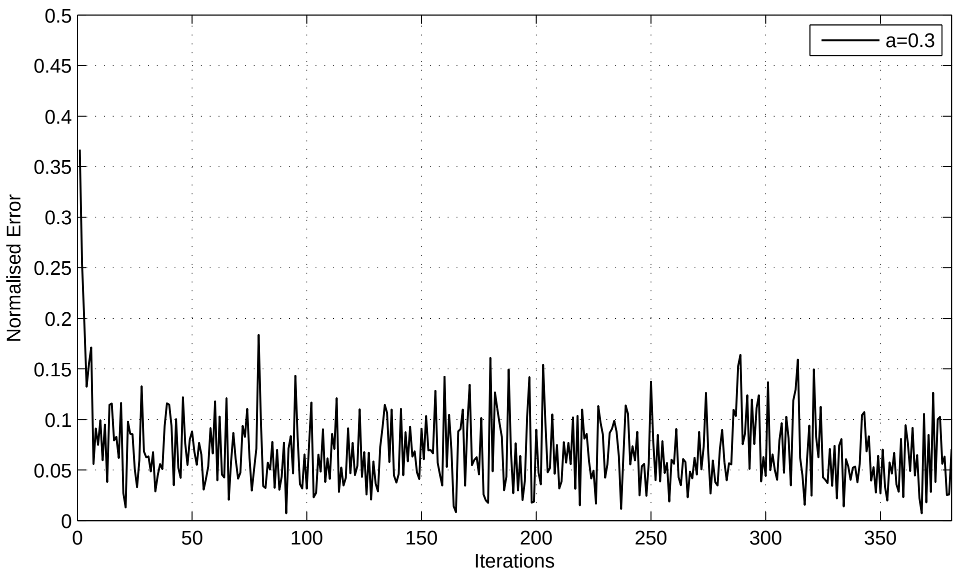

Fig. 6.4: Error signal with combined method at $\alpha = 0.3$.

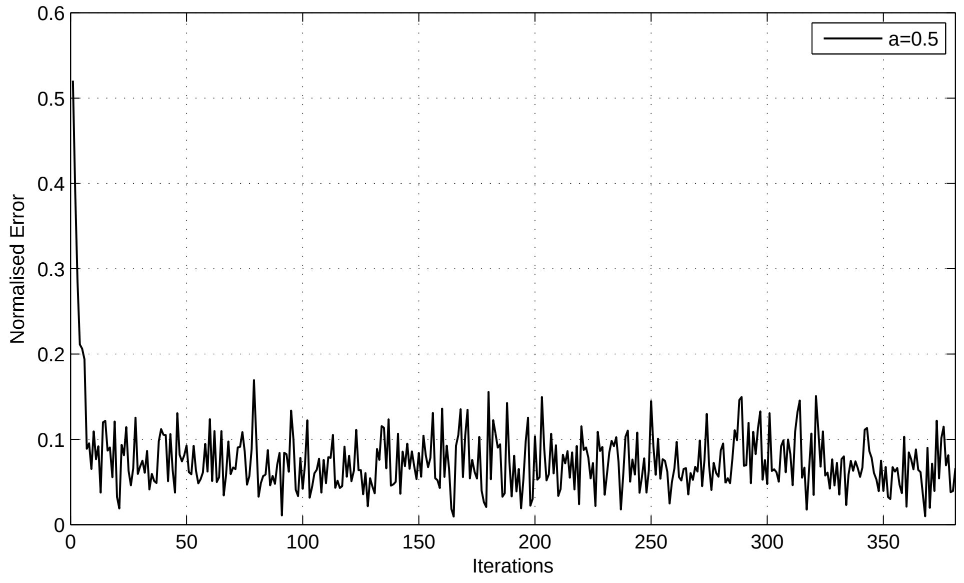

Fig. 6.5: Error signal with combined method at $\alpha = 0.5$.

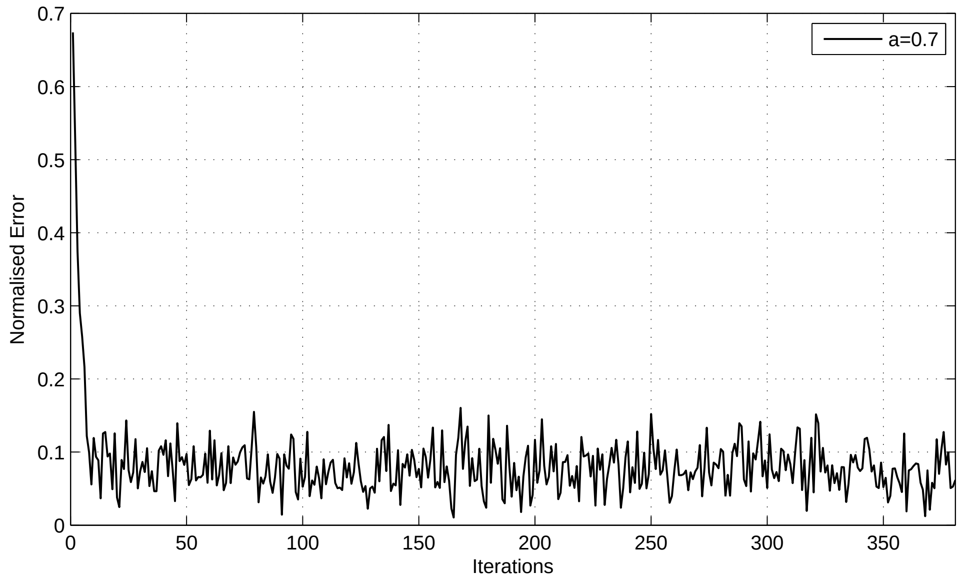

Fig. 6.6: Error signal with combined method at $\alpha = 0.7$.

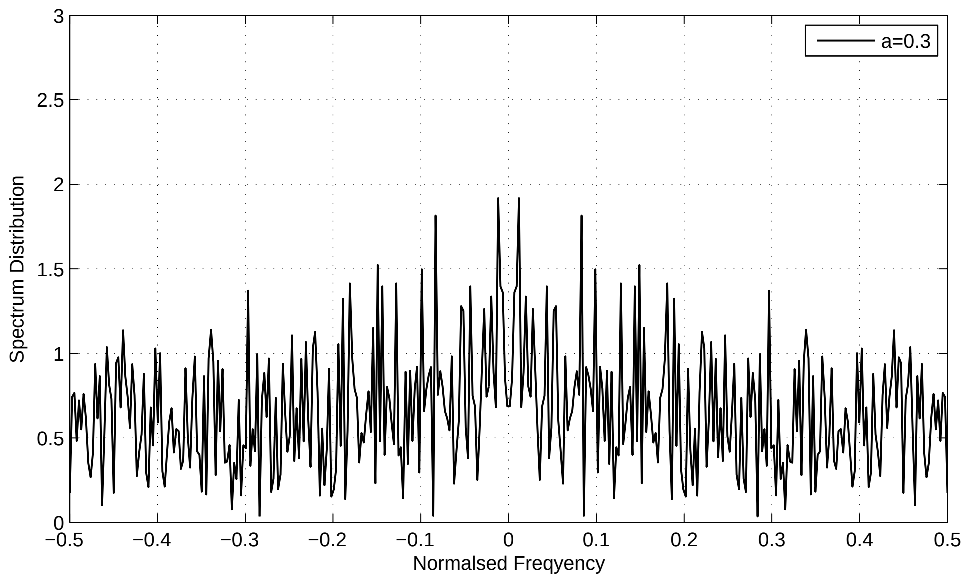

Fig. 6.7: The spectrum of error signal with combined method at $\alpha = 0.3$.

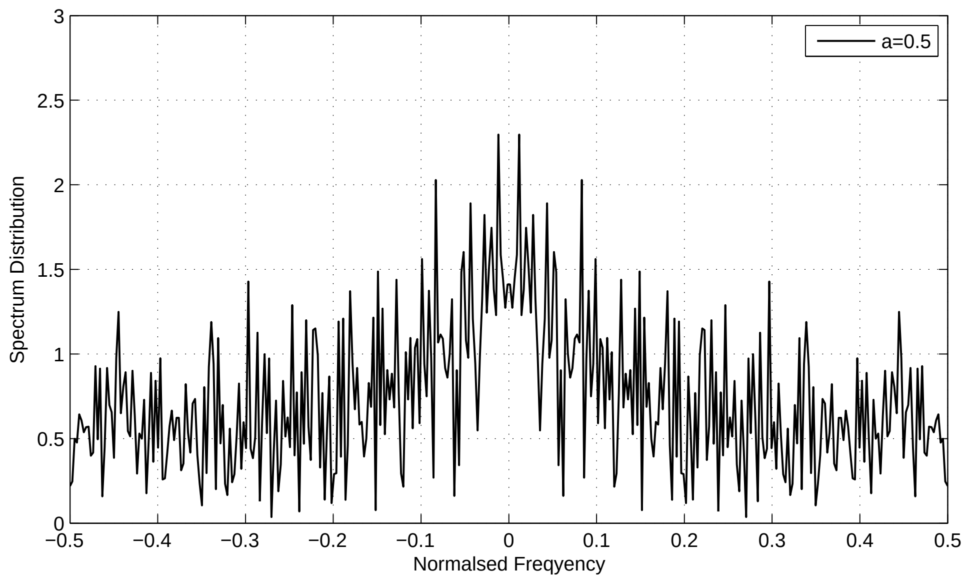

Fig. 6.8: The spectrum of error signal with combined method at $\alpha = 0.5$.

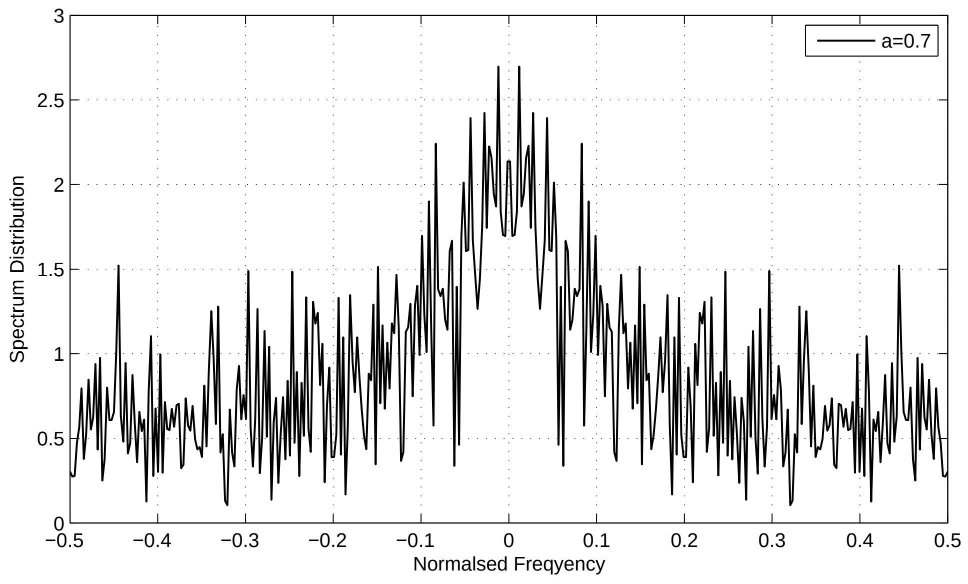

Fig. 6.9: The spectrum of error signal with combined method at α = 0.7.

algorithm is smaller. However, the LES has a higher computational complexity. As the computational time for LES is around 6 hours, which is much longer than the QLMS algorithm with only a few minutes, there is trade-off between accuracy and complexity.

6.3 Summary

In this chapter, the CFD approach has been reviewed briefly, including the CFD basic concepts and the simulation methods, and a new combined method is proposed by alternating the operation of the QLMS algorithm and the LES method in succesiion. As demonstrated by computer simulations, the proposed method has a much lower computational complexity with roughly half of the running time of a standard LES operation, while still maintaining a comparable performance in terms of prediction accuracy. A weighting method is also proposed, and its potential needs to be explored further in the future.

Chapter 7

Conclusions and Future Plan

7.1 Conclusions

In this thesis, the fundamental concept of quaternions was introduced in Chapter 2 and then the general quaternion-based gradient operator and its related chain rules as well as product rules were presented in detail in Chapter 3, where the QLMS algorithm and its augmented version the AQLMS algorithm are derived as an application together with a nonlinear adaptive algorithm.

Then, the application of quaternion-valued signal processing to adaptive beamforming based on crossed-dipole arrays was studied in Chapter 4, with the aim of reducing the number of sensors involved in the adaptive beamforming process, so that reduced system complexity and energy consumption can be achieved, whereas an acceptable performance can still be maintained. Such a technique is particularly useful for large array systems. In particular, based on the quaternion-valued steering vector model, a reweighted zero attracting QLMS algorithm was derived by introducing a RZA term to the cost function of the original QLMS algorithm. The RZA term aims to have a closer approximation to the l_0 norm so that the number of non-zero valued coefficients can be reduced more effectively in the adaptive beamforming process.

Another application to wind profile prediction was investigated in Chapter 5, where correlation analysis was provided to show that signal processing can provide a viable approach to wind profile prediction. Then, a quaternion-valued multi-channel frequency domain adaptive filtering structure was introduced, which can improve the convergence speed of the employed adaptive algorithms, leading to much better tracking capability for applications in a dynamic environment.

Base on the synergies of the signal processing approach and the CFD approach, a combined approach was proposed in Chapter 6. The basic concepts of CFD were first reviewed, including the direct numerical simulation (DNS) methods and the large eddy simulation (LES) methods. Then a combined method was proposed by alternating the operation of the QLMS algorithm and the LES method one by one. As demonstrated by computer simulations, the proposed method has a much lower computational complexity with roughly half of the running time of a standard LES operation, while still maintaining a comparable performance in terms of prediction accuracy. A weighting method is also proposed, and its potential needs to be further explored in the future.

7.2 Future Work

For future work, the focus will be further detailed investigation of the combined approach for wind profile prediction. Two combined methods have been proposed in the thesis: one is alternating and one is weighting. However, this part of the study is only preliminary and more details are needed. For example, for the alternating method, how frequent should the alternating be performed for more effective and efficient prediction? For the weighting method, what is the exact trade-off between complexity and accuracy and how to choose the best weighting function? How about combining the alternating method and the weighting method together to form a new combined method?

As an application of the developed combined methods, we can consider the wind farm design problem, which deals with the problem of finding the optimum wind turbine layout in terms of maximising energy production and minimising development costs, while meeting various geographical constraints imposed by the site.

A starting point in its design is effective modelling of atmospheric behaviour and interaction between wind and turbine. With constant moving of turbine blades, the modelling is a complicated process involving non-stationary boundary conditions. One possible approach is that we split it into a series of short periods of time, short enough so that we can consider the boundary is approximately stationary within this period of time. We can then apply some of our developed combined algorithms based on both signal processing and computational fluid dynamics to within each period of time and also to the transition between adjacent periods of time to smooth out the error. We may also need to split the whole three-dimensional space into small sub-spaces to deal with the problem more efficiently.

One particular focus for this part of the project is to reduce the computational complexity of the modelling process so that the problem can be solved not only effectively, but also efficiently. Based on the developed modelling algorithms, we then apply it to the problem of optimisation, analysis and design of wind farms, as given a specific layout of the wind farm, the atmospheric flow behaviour can be obtained for the whole farm and so the overall generated wind power. This is a highly nonlinear optimisation problem and various nonlinear optimisation algorithms will be employed in the process. At the end of the project, a software tool for wind farm design can be developed.

Bibliography

[1] X. Zhang, W. Liu, Y. Xu, and Z. Liu, “Quaternion-based worst case constrained beamformer based on electromagnetic vectoe-sensor arrays,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vancouver, Canada, May 2013, pp. 4149–6153.

[2] X. R. Zhang, W. Liu, Y. G. Xu, and Z. W. Liu, “Quaternion-valued robust adaptive beamformer for electromagnetic vector-sensor arrays with worst-case constraint,” *Signal Processing*, vol. 104, pp. 274–283, November 2014.

[3] M. D. Jiang, W. Liu, and Y. Li, “Adaptive beamforming for vector-sensor arrays based on reweighted zero-attracting quaternion-valued lms algorithm,” *IEEE Transations on Circuits and Systems II: Express Briefs*, vol. pp, 2016.

[4] N. Le Bihan and J. Mars, “Singular value decomposition of quaternion matrices: a new tool for vector-sensor signal processing,” *Signal Processing*, vol. 84, no. 7, pp. 1177–1199, 2004.

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

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

[7] M. D. Jiang, W. Liu, Y. Li, and X. R. Zhang, "Frequency-domain quaternion-valued adaptive filtering and its application to wind profile prediction," in *Proc. of the IEEE TENCON Conference*, Xi'an, China, October 2013.

[8] C. C. Took, G. Strbac, K. Aihara, and D. Mandic, "Quaternion-valued short-term joint forecasting of three-dimensional wind and atmospheric parameters," *Renewable Energy*, vol. 36, no. 6, pp. 1754–1760, 2011.

[9] S. Said, N. Le Bihan, and S. Sangwine, "Fast complexified quaternion fourier transform," *IEEE Transactions on Signal Processing*, vol. 56, no. 4, pp. 1522–1531, 2008.

[10] M. D. Jiang, W. Liu, and Y. Li, "A general quaternion-valued gradient operator and its applications to computational fluid dynamics and adaptive beamforming," in *Proc. of the International Conference on Digital Signal Processing*, Hong Kong, August 2014.

[11] X. M. Gou, Y. G. Xu, Z. W. Liu, and X. F. Gong, "Quaternion-Capon beamformer using crossed-dipole arrays," in *Proc. IEEE International Symposium on Microwave, Antenna, Propagation, and EMC Technologies for Wireless Communications (MAPE)*, November 2011, pp. 34–37.

[12] S. Miron, N. Le Bihan, and J. I. Mars, "Quaternion-MUSIC for vector-sensor array processing," *IEEE Transactions on Signal Processing*, vol. 54, no. 4, pp. 1218–1229, April 2006.

[13] N. Le Bihan, S. Miron, and J. I. Mars, "MUSIC algorithm for vector-sensors array using biquaternions," *IEEE Transactions on Signal Processing*, vol. 55, no. 9, pp. 4523–4533, 2007.

[14] J. W. Tao and W. X. Chang, "A novel combined beamformer based on hypercomplex processes," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 2, pp. 1276–1289, 2013.

[15] J. W. Tao, "Performance analysis for interference and noise canceller based on hypercomplex and spatio-temporal-polarisation processes," *IET Radar, Sonar Navigation*, vol. 7, no. 3, pp. 277–286, 2013.

[16] J. W. Tao and W. X. Chang, "Adaptive beamforming based on complex quaternion processes," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[17] M. B. Hawes and W. Liu, “Sparse vector sensor array design based on quaternionic formulations,” in *Proc. of the European Signal Processing Conference*, Lisbon, Portugal, September 2014.

[18] ——, “A quaternion-valued reweighted minimisation approach to sparse vector sensor array design,” in *Proc. of the International Conference on Digital Signal Processing*, Hong Kong, August 2014.

[19] Y. Chen, Y. Gu, and A. O. Hero, “Sparse LMS for system identification,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Taipei, April 2009, pp. 3125–3128.

[20] S. V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*, 2nd ed. New York: Wiley, 2002.

[21] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Englewood Cliffs, New York: Prentice Hall, 1996.

[22] W. Liu and S. Weiss, *Wideband Beamforming: Concepts and Techniques*. Chichester, UK: John Wiley & Sons, 2010.

[23] D. L. Duttweiler, “Proportionate normalized least-mean-squares adaptation in echo cancelers,” *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 508–518, 2000.

[24] B. Widrow, J. McCool, and M. Ball, “The Complex LMS Algorithm,” *Proceedings of the IEEE*, vol. 63, pp. 719–720, August 1975.

[25] T. A. Ell, N. Le Bihan, and S. J. Sangwine, *Quaternion Fourier transforms for signal and image processing*. John Wiley & Sons, 2014.

[26] S. Sangwine, “The discrete fourier transform of a colour image,” *Image Processing II Mathematical Methods, Algorithms and Applications*, pp. 430–441, 2000.

[27] M. Parfieniuk and A. Petrovsky, “Inherently lossless structures for eight- and six-channel linear-phase paraunitary filter banks based on quaternion multipliers,” *Signal Processing*, vol. 90, no. 6, pp. 1755–1767, 2010.

[28] H. Liu, Y. Zhou, and Z. Gu, “Inertial measurement unit-camera calibration based on incomplete inertial sensor information,” *Journal of Zhejiang University SCIENCE C*, vol. 15, no. 11, pp. 999–1008, 2014.

[29] M. Hawes and W. Liu, “Design of fixed beamformers based on vector-sensor arrays,” *International Journal of Antennas and Propagation*, vol. 2015, 2015.

[30] S. P. Talebi and D. P. Mandic, “A quaternion frequency estimator for three-phase power systems,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 3956–3960.

[31] W. Liu, “Antenna array signal processing for a quaternion-valued wireless communication system,” in *The Benjamin Franklin Symposium*

on Microwave and Antenna Sub-systems (BenMAS), Philadelphia, US, September 2014.

[32] W. R. Hamilton, *Elements of quaternions*. Longmans, Green, & co., 1866.

[33] T. A. Ell and S. J. Sangwine, "Quaternion involutions and anti-involutions," *Computers & Mathematics with Applications*, vol. 53, no. 1, pp. 137–143, 2007.

[34] R. Remmert, *Theory of complex functions*. Springer, 1991, vol. 122.

[35] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *IRE WESCON convention record*, vol. 4, no. 1. New York, 1960, pp. 96–104.

[36] A. H. Sayed, *Fundamentals of adaptive filtering*. John Wiley & Sons, 2003.

[37] B. Farhang-Boroujeny, *Adaptive filters: theory and applications*. John Wiley & Sons, 2013.

[38] S. Haykin, *Adaptive filter theory*, 4th ed. Prentice Hall, 2008.

[39] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, New York: Prentice Hall, 1985.

[40] W. Wirtinger, "Zur formalen theorie der funktionen von mehr komplexen veränderlichen," *Mathematische Annalen*, vol. 97, no. 1, pp. 357–376, 1926.

[41] A. Hjorungnes and D. Gesbert, "Complex-valued matrix differentiation: Techniques and key results," *IEEE Transactions on Signal Processing*, vol. 55, no. 6, pp. 2740–2746, 2007.

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

[43] C. Jahanchahi, C. C. Took, and D. P. Mandic, "On gradient calculation in quaternion adaptive filtering," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2012, pp. 3773–3776.

[44] Y. Xia, C. Jahanchahi, D. Xu, and D. Mandic, "The hc calculus, quaternion derivatives and caylay-hamilton form of quaternion adaptive filters and learning systems," in *2014 International Joint Conference on Neural Networks (IJCNN).* IEEE, July 2014, pp. 3395–3401.

[45] D. Xu, C. Jahanchahi, C. C. Took, and D. Mandic, "Quaternion derivatives: The ghr calculus," *arXiv preprint arXiv:1409.8168*, 2014.

[46] T. A. Ell, "Hypercomplex spectral transformations," Ph.D. dissertation, University of Minnesota, 1992.

[47] ——, "Quaternion-fourier transforms for analysis of two-dimensional linear time-invariant partial differential systems," in *Proceedings of the*

32nd IEEE Conference on Decision and Control. IEEE, 1993, pp. 1830–1841.

[48] S. J. Sangwine, “Fourier transforms of colour images using quaternion or hypercomplex, numbers,” *Electronics letters*, vol. 32, no. 21, pp. 1979–1980, 1996.

[49] S. Sangwine, “The discrete quaternion fourier transform,” in *Sixth International Conference on Image Processing and Its Applications*, vol. 2. IET, 1997, pp. 790–793.

[50] L. Shi, “Exploration in quaternion colour,” Ph.D. dissertation, School of Computing Science-Simon Fraser University, 2005.

[51] N. Le Bihan, S. Sangwine, and T. A. Ell, “Instantaneous frequency and amplitude of complex signals based on quaternion Fourier transform,” *ArXiv e-prints*, Aug. 2012.

[52] B. C. Ujang, C. C. Took, and D. Mandic, “Quaternion-valued nonlinear adaptive filtering,” *IEEE Transactions on Neural Networks*, vol. 22, no. 8, pp. 1193–1206, 2011.

[53] G. Gentili, and D. C. Struppa, “A new theory of regular functions of a quaternionic variable,” *Advances in Mathematics* , vol. 216, pp. 279–301, 2007.

[54] D. H. Brandwood, “A complex gradient operator and its application in adaptive array theory,” *IEE Proceedings H (Microwaves, Optics and Antennas)*, vol. 130, no. 1, pp. 11–16, 1983.

[55] C. C. Took and D. P. Mandic, “A quaternion widely linear adaptive filter,” *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 4427–4431, 2010.

[56] M. K. Roberts and R. Jayabalan, “An improved low-complexity sum-product decoding algorithm for low-density parity-check codes,” *Frontiers of Information Technology & Electronic Engineering*, vol. 16, pp. 511–518, 2015.

[57] B. Allen and M. Ghavami, *Adaptive Array Systems, Fundamentals and Applications*. Chichester, England: John Wiley & Sons, 2005.

[58] M. S. Brandstein and D. Ward, Eds., *Microphone Arrays: Signal Processing Techniques and Applications*. Berlin: Springer, 2001.

[59] N. Fourikis, *Advanced Array Systems, Applications and RF Technologies*. London: Academic Press, 2000.

[60] S. Haykin, *Array Signal Processing*, ser. Prentice-Hall Signal Processing Series. Englewood Cliffs: Prentice Hall, 1985.

[61] J. E. Hudson, *Adaptive Array Principles*, ser. IEE Electromagnetic Waves. London: The Institution of Electrical Engineers, 1981.

[62] D. H. Johnson and D. E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, ser. Signal Processing Series. Englewood Cliffs, NJ: Prentice Hall, 1993.

[63] R. A. Monzingo and T. W. Miller, *Introduction to Adaptive Arrays*. Raleigh, NC: SciTech Publishing Inc, 2004.

[64] H. L. Van Trees, *Optimum Array Processing, Part IV of Detection, Estimation, and Modulation Theory*. New York: Wiley, 2002.

[65] X. C. Chen, W. Zhang, W. Rhee, and Z. H. Wang, "A $\Delta\Sigma$ TDC-based beamforming method for vital sign detection radar systems," *IEEE Transactions on Circuits & Systems II: Express Briefs*, vol. 61, no. 12, pp. 932–936, December 2014.

[66] R. T. Compton, "The tripole antenna: An adaptive array with full polarization flexibility," *IEEE Transactions on Antennas and Propagation*, vol. 29, no. 6, pp. 944–952, November 1981.

[67] A. Nehorai, K. C. Ho, and B. T. G. Tan, "Minimum-noise-variance beamformer with an electromagnetic vector sensor," *IEEE Transactions on Signal Processing*, vol. 47, no. 3, pp. 601–618, March 1999.

[68] M. D. Zoltowski and K. T. Wong, "ESPRIT-based 2D direction finding with a sparse uniform array of electromagnetic vector-sensors," *IEEE Transactions on Signal Processing*, vol. 48, no. 8, pp. 2195–2204, August 2000.

[69] M. D. Jiang, W. Liu, and Y. Li, “A zero-attracting quaternion-valued least mean square algorithm for sparse system identification,” in *Proc. of IEEE/IET International Symposium on Communication Systems, Networks and Digital Signal Processing*, Manchester, UK, July 2014.

[70] J. Capon, “High-resolution frequency-wavenumber spectrum analysis,” *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, August 1969.

[71] B. Widrow, K. M. Duvall, R. P. Gooch, and W. C. Newman, “Signal cancellation phenomena in adaptive antennas: Causes and cures,” *IEEE Transactions on Antennas and Propagation*, vol. 30, no. 3, pp. 469–478, 1982.

[72] C. L. Zahm, “Application of adaptive arrays to suppress strong jammers in the presence of weak signals,” *IEEE Transactions on Aerospace and Electronic Systems*, no. 2, pp. 260–271, 1973.

[73] B. D. Van Veen and K. M. Buckley, “Beamforming: a versatile approach to spatial filtering,” *IEEE Acoustics, Speech, and Signal Processing Magazine*, vol. 5, no. 2, pp. 4–24, April 1988.

[74] B. Widrow, P. Mantey, L. Griffiths, and B. Goode, “Adaptive Antenna Systems,” *Proceedings of the IEEE*, vol. 55, pp. 2143–2159, December 1967.

[75] S. Applebaum, “Adaptive arrays,” *IEEE Transactions on Antennas and Propagation*, vol. 24, no. 5, pp. 585–598, 1976.

[76] R. T. Compton, *Adaptive Antennas: Concepts and Performance*. Englewood Cliffs, N.J.: Prentice Hall, 1988.

[77] V. Solo, “The Limiting Behaviour of LMS,” *IEEE Transactions on Signal Processing*, vol. 37, no. 12, pp. 1909–1922, December 1989.

[78] Z. Pritzker and A. Feuer, “Variable Length Stochastic Gradient Algorithm,” *IEEE Transactions on Signal Processing*, vol. SP-39, no. 4, pp. 997–1001, April 1991.

[79] R. Compton, “On the performance of a polarization sensitive adaptive array,” *IEEE Transactions on Antennas and Propagation*, vol. 29, no. 5, pp. 718–725, 1981.

[80] J. Li and R. Compton Jr, “Angle and polarization estimation using esprit with a polarization sensitive array,” *IEEE Transactions on Antennas and Propagation*, vol. 39, pp. 1376–1383, 1991.

[81] M. D. Jiang, Y. Li, and W. Liu, “Properties of a general quaternion-valued gradient operator and its application to signal processing,” *Frontiers of Information Technology & Electronic Engineering*, vol. 17, pp. 83–95, 2016.

[82] Q. Barthélemy, A. Larue, and J. I. Mars, “About QLMS derivations,” *IEEE Signal Processing Letters*, vol. 21, no. 2, pp. 240–243, 2014.

[83] J. Yoo, J. Shin, and P. Park, "An improved NLMS algorithm in sparse systems against noisy input signals," *IEEE Transactions on Circuits & Systems II: Express Briefs*, vol. 62, no. 3, pp. 271–275, March 2015.

[84] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, pp. 877–905, 2008.

[85] C. Rusu and B. Dumitrescu, "Iterative reweighted l 1 design of sparse fir filters," *Signal Processing*, vol. 92, no. 4, pp. 905–911, 2012.

[86] Y. H. Yang, W. P. Zhu, and D. L. Wu, "Design of sparse fir filters based on reweighted l 1-norm minimization," in *IEEE International Conference on Digital Signal Processing (DSP).* IEEE, 2015, pp. 858–862.

[87] B. Fuchs, "Synthesis of sparse arrays with focused or shaped beampattern via sequential convex optimizations," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 7, pp. 3499–3503, 2012.

[88] G. Prisco and M. D'Urso, "Maximally sparse arrays via sequential convex optimizations," *IEEE Antennas and Wireless Propagation Letters*, vol. 11, pp. 192–195, 2012.

[89] M. B. Hawes and W. Liu, "Sparse array design for wideband beamforming with reduced complexity in tapped delay-lines," *IEEE Transations on Audio, Speech and Language Processing*, vol. 22, pp. 1236–1247, August 2014.

[90] J. J. Shynk, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Processing Magazine*, vol. 9, pp. 14–37, January 1992.

[91] F. Beaufays, "Transform-Domain Adaptive Filters — An Analytical Approach," *IEEE Transactions on Signal Processing*, vol. 43, no. 2, pp. 422–431, February 1995.

[92] Y. Chen and H. Fang, "Frequency-domain implementation of griffiths-jim adaptive beamformer," *Journal of the Acoustic Society of America*, vol. 91, pp. 3354–3366, June 1992.

[93] J. An and B. Champagne, "GSC realisations using the two-dimensional transform LMS algorithm," *IEE Proc. Radar, Sonar and Navig.*, vol. 141, pp. 270–278, October 1994.

[94] D. Mandic, S. Javidi, S. Goh, A. Kuh, and K. Aihara, "Complex-valued prediction of wind profile using augmented complex statistics," *Renewable Energy*, vol. 34, no. 1, pp. 196–201, 2009.

[95] W. Liu, S. Weiss, and L. Hanzo, "Low-complexity frequency-domain GSC for broadband beamforming," in *Proc. International Conference on Signal Processing*, vol. 1, Beijing, China, August 2002, pp. 386–389.

[96] Google, "Surface level wind data collection, http://code.google.com/p/google-rec-csp/downloads/list."

[97] J. Ferzige and M. Peric, *Computational methods for fluid dynamics*. Springer, 2001.

[98] H. Tennekes and J. L. Lumley, *A first course in turbulence*. MIT press, 1972.

[99] S. B. Pope, *Turbulent flows*. Cambridge university press, 2000.

[100] G. Evans, J. Blackledge, and P. Yardley, *Numerical methods for partial differential equations*. Springer Science & Business Media, 2012.

[101] C. Canuto, A. Quarteroni, M. Hussaini, and T. Zang, "Spectral method in fluid mechanics," 1988.

[102] S. Orszag, "Analytical theories of turbulence," *Journal of Fluid Mechanics*, vol. 41, no. 02, pp. 363–386, 1970.

[103] J. Smagorinsky, "General circulation experiments with the primitive equations: I. the basic experiment*," *Monthly weather review*, vol. 91, no. 3, pp. 99–164, 1963.

[104] J. Deardorff, "A numerical study of three-dimensional turbulent channel flow at large reynolds numbers," *Journal of Fluid Mechanics*, vol. 41, no. 02, pp. 453–480, 1970.