University of Reading

Zero attracting recursive least squares algorithms

Article

Accepted Version

Hong, X., Gao, J. and Chen, S. (2017) Zero attracting recursive least squares algorithms. IEEE Transactions on Vehicular Technology, 66 (1). 213 -221. ISSN 0018-9545 doi: https://doi.org/10.1109/TVT.2016.2533664 Available at https://centaur.reading.ac.uk/65511/

It is advisable to refer to the publisher's version if you intend to cite from the work. See Guidance on citing.

To link to this article DOI: http://dx.doi.org/10.1109/TVT.2016.2533664

Publisher: IEEE

All outputs in CentAUR are protected by Intellectual Property Rights law, including copyright law. Copyright and IPR is retained by the creators or other copyright holders. Terms and conditions for use of this material are defined in the End User Agreement.

www.reading.ac.uk/centaur

CentAUR

Central Archive at the University of Reading

Reading's research outputs online

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

Zero Attracting Recursive Least Squares Algorithms

Xia Hong, *Senior Member, IEEE,* Junbin Gao, Sheng Chen, *Fellow, IEEE*

***Abstract*—The l_1-norm sparsity constraint is a widely used technique for constructing sparse models. In this contribution, two zero-attracting recursive least squares algorithms, referred to as ZA-RLS-I and ZA-RLS-II, are derived by employing the l_1-norm of parameter vector constraint to facilitate the model sparsity. In order to achieve a closed-form solution, the l_1-norm of the parameter vector is approximated by an adaptively weighted l_2-norm, in which the weighting factors are set as the inversion of the associated l_1-norm of parameter estimates that are readily available in the adaptive learning environment. ZA-RLS-II is computationally more efficient than ZA-RLS-I by exploiting the known results from linear algebra as well as the sparsity of the system. The proposed algorithms are proven to converge, and adaptive sparse channel estimation is used to demonstrate the effectiveness of the proposed approach.**

***Index Terms*—Zero attracting recursive least squares algorithm, adaptive channel estimation, sparse model, l_1-norm**

I. Introduction

Adaptive filtering and system identification algorithms [1], e.g. the least mean squares (LMS), the normalized least mean squares (NLMS) and the recursive least squares (RLS) algorithms, are widely used in estimation problems such as channel estimation. In communications, the multipath wireless channel is characterized by multipath taps that are widely spread in time, with only a few significant components. Intuitively this inherent sparsity of the channel impulse response (CIR) should be exploited to improve the quality of channel estimation. However neither RLS nor LMS exploits the underlying sparsity in the data process and their achievable system performance can be seriously impaired.

Alternatively the sparse representation of an observed signal, in which the given signal is modeled as a linear combination of some significant atoms taken from an over-complete dictionary, has been widely researched in the areas of computational biology, medicine, neuroscience, and compressive sensing. With regard to sparse modeling for a given dictionary, many algorithms exist which are divided into three categories: optimization-based methods, greedy-based methods and thresholding-based methods. Basis pursuit (BP) is a commonly used optimization method, which uses a convex optimization method to minimize the l_1 norm of the sparse coefficient vector [2], [3]. The computational complexity of the BP is very high and, therefore, it is not suitable for large-scale problems. In comparison, the matching pursuit (MP), a greedy algorithm, has a significantly lower complexity than the BP, especially when the sparsity level is low [4]. A popular extension of MP is the orthogonal matching pursuit (OMP) [5], [6], which iteratively refines a sparse representation by successively identifying one atom at a time that yields the greatest improvement in modeling quality until an expected sparsity level is achieved or the approximation error is below the given threshold. The thresholding-based methods contain algorithms that do not require an estimation of the sparsity. In these algorithms, the hard thresholding operator gives way to a soft thresholding operator with a positive threshold, such as in the iterative hard thresholding (IHT) algorithm [7] and the hard thresholding pursuit (HTP) [8]. Another important sparse modeling method is the message-passing algorithm studied in [9]. Unfortunately, all the above-mentioned algorithms are not designed for time-varying environments, such as vehicular communication applications, and thus they are not appropriate for the problem of sparse channel estimation in real time.

The LMS algorithm is one of the most popular adaptive algorithms for channel estimation since it has a very low computational complexity and is easy to implement at the mobile handset receiver. Several adaptive sparse modeling algorithms have been proposed based on LMS recently [10]–[13]. A good example of them is the zero-attracting LMS (ZA-LMS) algorithm proposed in [10]. This algorithm can achieve a faster convergence rate, while reducing the steady-state excess mean square error (MSE), compared to the classic LMS algorithm. The ZA-LMS algorithm introduces an l_1-norm of the parameter vector in the cost function of the LMS algorithm, which modifies the parameter vector update equation with a zero attractor term. Similarly, the zero-attracting NLMS (ZA-NLMS) algorithm has been introduced based on NLMS, which yields better performance than the NLMS [12]. The l_0-norm, which is defined as the number of non-zero terms in the parameter vector, is a more appropriate measure of sparsity, and the work of [13] introduces an l_0-norm of the parameter vector in the cost function of the LMS algorithm. However, a nonlinear approximation to l_0-norm is needed in practical implementation, and this requires an additional tuning parameter [12].

Recently sparse solutions have been proposed based on RLS algorithms [14]–[21]. The so-called SPARLS algorithm is introduced in [14] using an expectation-maximization (EM) approach. The work of [15] proposes an adaptive version of the greedy least squares method using partial orthogonalization to systems. The work of [16] modifies the RLS algorithm by using a general convex function of the system parameters, resulting in the l_0-RLS and l_1-RLS algorithms. In comparison to the LMS based algorithms, the RLS based algorithms have

Copyright ©2016 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Xia Hong is with Department of Computer Science, School of Mathematical and Physical Sciences, University of Reading, Reading, RG6 6AY, UK (E-mail: x.hong@reading.ac.uk).

Junbin Gao is with Discipline of Business Analytics, The University of Sydney Business School, The University of Sydney, Camperdown NSW 2006, Australia (E-mail: junbin.gao@sydney.edu.au).

Sheng Chen is with Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK (E-mail: sqc@ecs.soton.ac.uk). He is also with King Abdulaziz University, Jeddah 21589, Saudi Arabia.

This research was partly supported by Australian Research Council (ARC) through the grant DP140102270.

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

a faster convergence speed as well as yield more accurate parameter estimates.

Against this background, in this contribution we propose two zero-attracting RLS (ZA-RLS) algorithms, referred to as ZA-RLS-I and ZA-RLS-II. Similar to [10], [16], the l_1-norm of the parameter vector penalty is added to the RLS cost function. For tractability, we further approximate the l_1-norm of the parameter vector penalty term as an adaptively weighted l_2-norm of the parameter vector term, in which the weights are readily given by the inversion of the associated l_1-norm of the parameter estimates that are currently available in the adaptive learning environment. We initially derive the ZA-RLS-I which however has a higher computational cost than the RLS algorithm, due to the need for a matrix inversion at each adapting step. In order to overcome this limitation, the ZA-RLS-II is then designed which has a computational cost comparable to the RLS algorithm, and this is achieved by exploiting matrix theory and structural properties of the matrices involved. Additionally, an analysis on the convergence of the proposed algorithms is given. Sparse channel identification results are included to demonstrate that our ZA-RLS approach achieves better performance, in comparison to the existing l_1-RLS algorithm of [16] and SPARLS algorithm of [14].

Throughout this contribution, $(\ )^*$ denotes complex conjugate, while $(\ )^{\mathrm{T}}$ and $(\ )^{\mathrm{H}}$ denote the vector or matrix transpose and Hermitian operators, respectively. $(\ )^{-1}$ stands for the inverse operation and the expectation operator is denoted by $\mathrm{E}\{\ \}$. Furthermore, $\boldsymbol{I}$ denotes the identity matrix with an appropriate dimension, and $\mathrm{diag}\{d_0, d_1, \cdots, d_L\}$ is the diagonal matrix with $d_0, d_1, \cdots, d_L$ as its diagonal elements, while $\mathrm{tr}\{\ \}$ denotes the matrix trace operation.

II. Adaptive algorithms with l_1-norm sparsity

Consider a transmission link that is represented by a finite-duration impulse response (FIR) filter of order L. It is assumed that the channel is inherently sparse in which only a few CIR coefficients are dominant with large values, but most of the CIR taps are zero or close to zero. Given the input signal $x(k) \in \mathbb{C}$, the received output signal $y(k) \in \mathbb{C}$ is described by

$$y(k) = \sum_{i=0}^{L} h_i x(k-i) + n(k) = \boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{h} + n(k), \quad (1)$$

where k denotes the symbol index and $n(k) \in \mathbb{C}$ is the channel additive white Gaussian noise (AWGN) with power of $N_o = \mathrm{E}\{|n(k)|^2\} = \mathrm{E}\{n(k)n^*(k)\} = 2\sigma_n^2$, while $\boldsymbol{h} = \begin{bmatrix} h_0 & h_1 \cdots h_L \end{bmatrix}^{\mathrm{T}}$ denotes the CIR coefficient vector and $\boldsymbol{x}(k) = [x(k)\ x(k-1) \cdots x(k-L)]^{\mathrm{T}}$ is the input vector.

In adaptive filtering and system identification, the parameters are computed recursively in time, so that the estimate $\widehat{\boldsymbol{h}}(k)$, as an estimate of $\boldsymbol{h}$ at time k, is given as a modification of $\widehat{\boldsymbol{h}}(k-1)$, based on the error signal $e(k) = y(k) - \boldsymbol{x}^{\mathrm{T}}(k)\widehat{\boldsymbol{h}}(k-1)$ upon the arrival of the new data $\{x(k), y(k)\}$. Note that the model sparsity can be achieved by adding an l_1-norm penalty to its parameters in the cost function, and here we briefly review several l_1-norm based adaptive algorithms which will be used in our comparative studies.

The ZA-LMS algorithm: It is proposed in [10] to minimizing the cost function given by

$$V_{ZA-LMS}\big(\widehat{\boldsymbol{h}}\big) = \frac{1}{2}|e(k)|^2 + \rho_{ZA-LMS}\sum_{i=0}^{L}\big|\widehat{h}_i\big|, \quad (2)$$

where $\rho_{ZA-LMS} > 0$ is a small regularization parameter. It can be seen that the cost function (2) is obtained by adding the l_1 norm of the parameter vector, $\widehat{\boldsymbol{h}} = \begin{bmatrix}\widehat{h}_0\ \widehat{h}_1 \cdots \widehat{h}_L\end{bmatrix}^{\mathrm{T}}$, to the LMS cost function based on the instantaneous squared error, $\frac{1}{2}|e(k)|^2$. This results in the following simple update equation for the ZA-LMS algorithm [10]

$$\begin{aligned}\widehat{\boldsymbol{h}}(k) =& \widehat{\boldsymbol{h}}(k-1) + \mu \cdot \boldsymbol{x}^*(k)e(k) \\ & - \mu \cdot \rho_{ZA-LMS} \cdot \mathrm{sgn}\big(\widehat{\boldsymbol{h}}(k-1)\big), \end{aligned} \quad (3)$$

where $\mu > 0$ is a preset small learning rate parameter and $\mathrm{sgn}(u)$ is the component-wise sign function defined by

$$\mathrm{sgn}(u) = \begin{cases} \dfrac{u}{|u|}, & \text{if } u \neq 0, \\ 0, & \text{if } u = 0. \end{cases}$$

The algorithmic parameters in ZA-LMS can be set using the criteria proposed in [22], [23]. Specifically, these are based on steady-state mean squares deviation convergence analysis using white input signal [22] and application of LMS to a distributed network [22], respectively.

The ZA-NLMS algorithm: The ZA-NLMS algorithm [12] is given as

$$\begin{aligned}\widehat{\boldsymbol{h}}(k) =& \widehat{\boldsymbol{h}}(k-1) + \mu \cdot \frac{\boldsymbol{x}^*(k)e(k)}{\boldsymbol{x}^{\mathrm{H}}(k)\boldsymbol{x}(k)} \\ & - \mu \cdot \rho_{ZA-NLMS} \cdot \mathrm{sgn}\big(\widehat{\boldsymbol{h}}(k-1)\big), \end{aligned} \quad (4)$$

where $\rho_{ZA-NLMS} > 0$ is a small regularization parameter. The works [22], [23] can be extended to ZA-NLMS in selecting its algorithmic parameters.

The l_1-RLS algorithm: The l_1-RLS algorithm [16] is based on minimizing the following cost function

$$V\big(\widehat{\boldsymbol{h}}\big) = \sum_{s=1}^{k}\lambda^{k-s}|e(k)|^2 + \rho\sum_{i=0}^{L}\big|\widehat{h}_i\big|, \quad (5)$$

where λ is a forgetting factor that is slightly less than 1, in the range of 0.95 to 0.99, and $\rho > 0$ is a small regularization parameter. The parameters are updated using the following equations:

$$\begin{cases} e(k) = y(k) - \boldsymbol{x}^{\mathrm{T}}(k)\widehat{\boldsymbol{h}}(k-1), \\ \boldsymbol{P}(k) = \dfrac{1}{\lambda}\left(\boldsymbol{P}(k-1) - \dfrac{\boldsymbol{P}(k-1)\boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{P}(k-1)}{\lambda + \boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{P}(k-1)\boldsymbol{x}^*(k)}\right), \\ \widehat{\boldsymbol{h}}(k) = \widehat{\boldsymbol{h}}(k-1) + \boldsymbol{P}(k)\boldsymbol{x}^*(k)e(k) \\ \qquad -\rho(1-\lambda)\boldsymbol{P}(k)\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(k-1)\big), \end{cases}$$

where $\widehat{\boldsymbol{h}}(0)$ is initialized as a zero or small random vector, and $\boldsymbol{P}(k)$ is the covariance matrix that is initialized to $\boldsymbol{P}(0) = \frac{1}{\delta}\boldsymbol{I}$ with δ being a very small positive number.

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

The SPARLS algorithm: The SPARLS algorithm [14] also uses an l_1 penalty, which modifies a wavelet based image restoration algorithm [24] to adaptive filtering setting. It is a probability modeling approach based on Gaussian noise assumption and the penalized maximum likelihood estimation. Since this penalized maximum likelihood estimation problem is hard to solve, the idea of [24] is to decompose the noise into a sum of two Gaussian components which leads to the use of iterative EM algorithm as the solver. The advantage of the SPARLS algorithm is that it has guaranteed theoretical convergence. The details of this algorithm can be found in [14].

III. The Proposed Zero Attracting Recursive Least Squares Algorithms

We initially introduce the ZA-RLS-I by modifying the conventional RLS to include the l_1-norm sparsity constraint. The ZA-RLS-II is then proposed to improve computational efficiency.

A. ZA-RLS-I

Like the l_1-RLS algorithm of [16], the two proposed ZA-RLS algorithms are also based on the cost function (5). However, since the cost function (5) does not lend to a closed-form solution, we use the strategy of relaxation by approximating (5) as

$$V_{ZA-RLS}\big(\widehat{\boldsymbol{h}}\big) = \sum_{s=1}^{k} \lambda^{k-s}|e(k)|^2 + \rho \cdot \widehat{\boldsymbol{h}}^{\mathrm{H}} \boldsymbol{D}(k)\widehat{\boldsymbol{h}}, \tag{6}$$

where $\boldsymbol{D}(k) = \mathrm{diag}\{d_0(k), d_1(k), \cdots, d_L(k)\}$ with $d_i(k) = \dfrac{1}{\big|\widehat{h}_i(k-1)\big| + \epsilon}$ for $0 \le i \le L$, while $\epsilon > 0$ is a very small positive number, e.g. $\epsilon = 10^{-7}$, which is introduced for numerical stability reason. Denote $\boldsymbol{y}(k) = \big[y(1)\ y(2) \cdots y(k)\big]^{\mathrm{T}}$, $\boldsymbol{\Lambda}(k) = \mathrm{diag}\{\lambda^{k-1}, \cdots, \lambda, 1\}$, and

$$\boldsymbol{X}(k) = \left[\begin{array}{c} \boldsymbol{X}(k-1) \\ \boldsymbol{x}^{\mathrm{T}}(k) \end{array}\right]$$

with $\boldsymbol{X}(1) = \boldsymbol{x}^{\mathrm{T}}(1) = [x(1)\ 0 \cdots 0]$. The cost function (6) can be equivalently represented as

$$\begin{aligned} V_{ZA-RLS}\big(\widehat{\boldsymbol{h}}\big) =& \big(\boldsymbol{y}(k) - \boldsymbol{X}(k)\widehat{\boldsymbol{h}}\big)^{\mathrm{H}} \boldsymbol{\Lambda}(k) \big(\boldsymbol{y}(k) - \boldsymbol{X}(k)\widehat{\boldsymbol{h}}\big) \\ &+ \rho \cdot \widehat{\boldsymbol{h}}^{\mathrm{H}} \boldsymbol{D}(k)\widehat{\boldsymbol{h}}. \end{aligned} \tag{7}$$

The minimizer of (7) is given by

$$\widehat{\boldsymbol{h}}(k) = \boldsymbol{P}(k)\boldsymbol{X}^{\mathrm{H}}(k)\boldsymbol{\Lambda}(k)\boldsymbol{y}(k), \tag{8}$$

where $\boldsymbol{P}(k) = \Big(\boldsymbol{X}^{\mathrm{H}}(k)\boldsymbol{\Lambda}(k)\boldsymbol{X}(k) + \rho\boldsymbol{D}(k)\Big)^{-1}$. At time index $(k-1)$, (8) is in the form of

$$\widehat{\boldsymbol{h}}(k-1) = \boldsymbol{P}(k-1)\boldsymbol{X}^{\mathrm{H}}(k-1)\boldsymbol{\Lambda}(k-1)\boldsymbol{y}(k-1). \tag{9}$$

It is easy to verify that

$$\begin{aligned} \boldsymbol{P}^{-1}(k) =& \lambda\boldsymbol{P}^{-1}(k-1) + \boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k) \\ &+ \rho(\boldsymbol{D}(k) - \lambda\boldsymbol{D}(k-1)) \end{aligned} \tag{10}$$

and

$$\begin{aligned} \boldsymbol{X}^{\mathrm{H}}(k)\boldsymbol{\Lambda}(k)\boldsymbol{y}(k) =& \lambda\boldsymbol{X}^{\mathrm{H}}(k-1)\boldsymbol{\Lambda}(k-1)\boldsymbol{y}(k-1) \\ &+ \boldsymbol{x}^*(k)y(k). \end{aligned} \tag{11}$$

Substituting (9) and (10) into (11) and noting $e(k) = y(k) - \boldsymbol{x}^{\mathrm{T}}(k)\widehat{\boldsymbol{h}}(k-1)$ yield

$$\begin{aligned} &\boldsymbol{X}^{\mathrm{H}}(k)\boldsymbol{\Lambda}(k)\boldsymbol{y}(k) = \lambda\boldsymbol{P}^{-1}(k-1)\widehat{\boldsymbol{h}}(k-1) + \boldsymbol{x}^*(k)y(k) \\ &= \Big(\boldsymbol{P}^{-1}(k) - \boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k) - \rho(\boldsymbol{D}(k) - \lambda\boldsymbol{D}(k-1))\Big)\widehat{\boldsymbol{h}}(k-1) \\ &\quad + \boldsymbol{x}^*(k)y(k) \\ &= \boldsymbol{P}^{-1}(k)\widehat{\boldsymbol{h}}(k-1) - \rho(\boldsymbol{D}(k) - \lambda\boldsymbol{D}(k-1))\widehat{\boldsymbol{h}}(k-1) \\ &\quad + \boldsymbol{x}^*(k)e(k). \end{aligned} \tag{12}$$

Substituting (12) into (8) leads to the recursive formula for updating the parameter vector:

$$\begin{aligned} \widehat{\boldsymbol{h}}(k) =& \widehat{\boldsymbol{h}}(k-1) - \rho\boldsymbol{P}(k)(\boldsymbol{D}(k) - \lambda\boldsymbol{D}(k-1))\widehat{\boldsymbol{h}}(k-1) \\ &+ \boldsymbol{P}(k)\boldsymbol{x}^*(k)e(k). \end{aligned} \tag{13}$$

We now derive the recursive formula for calculating $\boldsymbol{P}(k)$. Note that by defining $\boldsymbol{Q}^{-1}(k) = \boldsymbol{P}^{-1}(k) - \rho\boldsymbol{D}(k)$, Eq. (10) is equivalent to

$$\boldsymbol{Q}^{-1}(k) = \lambda\boldsymbol{Q}^{-1}(k-1) + \boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k). \tag{14}$$

Using the famous matrix inversion lemma, we have

$$\boldsymbol{Q}(k) = \frac{1}{\lambda}\left(\boldsymbol{Q}(k-1) - \frac{\boldsymbol{Q}(k-1)\boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{Q}(k-1)}{\lambda + \boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{Q}(k-1)\boldsymbol{x}^*(k)}\right). \tag{15}$$

On the other hand, if we apply the matrix inversion lemma based on

$$\boldsymbol{P}^{-1}(k) = \boldsymbol{Q}^{-1}(k) + \rho\boldsymbol{D}(k), \tag{16}$$

we have

$$\boldsymbol{P}(k) = \boldsymbol{H}(k) - \boldsymbol{H}(k)\big(\boldsymbol{H}(k) + \boldsymbol{Q}(k)\big)^{-1}\boldsymbol{H}(k), \tag{17}$$

where

$$\begin{aligned} \boldsymbol{H}(k) =& \big(\rho\boldsymbol{D}(k)\big)^{-1} = \mathrm{diag}\Big\{\big(\big|\widehat{h}_0(k-1)\big| + \epsilon\big)/\rho, \\ &\cdots, \big(\big|\widehat{h}_L(k-1)\big| + \epsilon\big)/\rho\Big\}. \end{aligned} \tag{18}$$

We summarize the proposed ZA-RLS-I algorithm in Algorithm 1. Clearly, the ZA-RLS-I algorithm has a higher computational cost than the standard RLS, owing to the fact that the matrix inversion is still needed to calculate $\boldsymbol{P}(k)$. When $\rho = 0$, it reduce to the conventional RLS algorithm since it can be shown that $\boldsymbol{P}(k) = \boldsymbol{Q}(k)\Big(\boldsymbol{H}(k) + \boldsymbol{Q}(k)\Big)^{-1}\boldsymbol{H}(k)$ in which the term $\Big(\boldsymbol{H}(k) + \boldsymbol{Q}(k)\Big)^{-1}\boldsymbol{H}(k)$ tends to the identity matrix.

B. ZA-RLS-II

We now propose a more efficient version of ZA-RLS, referred to as the ZA-RLS-II algorithm, by exploiting the structural properties of the matrices involved. Note that by denoting $\boldsymbol{Q}^{[inv]}(k) = \boldsymbol{Q}^{-1}(k)$, Eq. (14) can be alternatively represented by

$$\boldsymbol{Q}^{[inv]}(k) = \lambda\boldsymbol{Q}^{[inv]}(k-1) + \boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k). \tag{19}$$

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

Algorithm 1 ZA-RLS-I algorithm.

1: Initialize $\widehat{\boldsymbol{h}}(0)$ as a zero or small random vector. Set $\boldsymbol{Q}(0)=\frac{1}{\delta}\boldsymbol{I}$ with δ being a very small positive number. Initialize both $\boldsymbol{D}(0)$ and $\boldsymbol{D}(1)$ as zero matrix.
2: **for** time step $k=1,2,\cdots$, **do**
3: Calculate

$$\left\{\begin{array}{rcl} e(k) &=& y(k)-\boldsymbol{x}^{\mathrm{T}}(k)\widehat{\boldsymbol{h}}(k-1)\\ \boldsymbol{H}(k) &=& \mathrm{diag}\Big\{\big(\big|\widehat{h}_0(k-1)\big|+\epsilon\big)/\rho,\cdots,\big(\big|\widehat{h}_L(k-1)\big|+\epsilon\big)/\rho\Big\}\\ \boldsymbol{D}(k) &=& \mathrm{diag}\left\{\dfrac{1}{\big|\widehat{h}_0(k-1)\big|+\epsilon},\cdots,\dfrac{1}{\big|\widehat{h}_L(k-1)\big|+\epsilon}\right\},\ \text{for}\ \ k>1\\ \boldsymbol{Q}(k) &=& \dfrac{1}{\lambda}\left(\boldsymbol{Q}(k-1)-\dfrac{\boldsymbol{Q}(k-1)\boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{Q}(k-1)}{\lambda+\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{Q}(k-1)\boldsymbol{x}^*(k)}\right)\\ \boldsymbol{P}(k) &=& \boldsymbol{H}(k)-\boldsymbol{H}(k)\Big(\boldsymbol{Q}(k)+\boldsymbol{H}(k)\Big)^{-1}\boldsymbol{H}(k)\\ \widehat{\boldsymbol{h}}(k) &=& \widehat{\boldsymbol{h}}(k-1)-\rho\boldsymbol{P}(k)\big(\boldsymbol{D}(k)-\lambda\boldsymbol{D}(k-1)\big)\widehat{\boldsymbol{h}}(k-1)+\boldsymbol{P}(k)\boldsymbol{x}^*(k)e(k)\end{array}\right.$$

4: **end for**

The basic idea of the ZA-RLS-II algorithm is to calculate $\boldsymbol{S}(k)=\big(\boldsymbol{H}(k)+\boldsymbol{Q}(k)\big)^{-1}$ in terms of $\boldsymbol{Q}^{[inv]}(k)$ which can easily be updated recursively. The ZA-RLS-II algorithm also makes use of the sparsity property of the channel. Specifically, we note that as a result of the sparse channel, $\boldsymbol{H}(k)$ has a low rank, i.e. it is a diagonal matrix with only a few dominant components. We start with introducing a mathematical theorem [25].

Lemma 1 ([25]): If $\boldsymbol{Q}$ and $\boldsymbol{H}$ are nonsingular square matrices of the same dimension, and $\boldsymbol{H}$ has rank one, then

$$\big(\boldsymbol{Q}+\boldsymbol{H}\big)^{-1}=\boldsymbol{Q}^{-1}-\frac{1}{1+g}\boldsymbol{Q}^{-1}\boldsymbol{H}\boldsymbol{Q}^{-1}, \tag{20}$$

where $g=\mathrm{tr}\big\{\boldsymbol{H}\boldsymbol{Q}^{-1}\big\}$.

Theorem 1 ([25]): Let $\boldsymbol{Q}$ and $\boldsymbol{H}$ be nonsingular square matrices of the same dimension. Suppose that $\boldsymbol{H}$ has rank $r>0$ and is decomposed as $\boldsymbol{H}=\sum_{i=1}^{r}\boldsymbol{H}_i$, where each $\boldsymbol{H}_i$ has rank one. Denote $\boldsymbol{Q}_{i+1}=\boldsymbol{Q}+\sum_{j=1}^{i}\boldsymbol{H}_j$ with $\boldsymbol{Q}_1=\boldsymbol{Q}$. Then by making use of Lemma 1, we have

$$\boldsymbol{Q}_{i+1}^{-1}=\boldsymbol{Q}_i^{-1}-\frac{1}{1+g_i}\boldsymbol{Q}_i^{-1}\boldsymbol{H}_i\boldsymbol{Q}_i^{-1}, \tag{21}$$

where $g_i=\mathrm{tr}\big\{\boldsymbol{H}_i\boldsymbol{Q}_i^{-1}\big\}$. In particular,

$$\big(\boldsymbol{Q}+\boldsymbol{H}\big)^{-1}=\boldsymbol{Q}_{r+1}^{-1}=\boldsymbol{Q}_r^{-1}-\frac{1}{1+g_r}\boldsymbol{Q}_r^{-1}\boldsymbol{H}_r\boldsymbol{Q}_r^{-1}. \tag{22}$$

Consider applying Theorem 1 to $\boldsymbol{S}(k)=\big(\boldsymbol{H}(k)+\boldsymbol{Q}(k)\big)^{-1}$, by decomposing $\boldsymbol{H}(k)$ as a series of r rank-one matrices, where r is the number of nonzero taps in $\widehat{\boldsymbol{h}}(k-1)$. Specifically, at each time step k, we find the integer set ω as

$$\omega=\big\{i|0\le i\le L,|\widehat{h}_i(k-1)|>\xi\big\}, \tag{23}$$

where ξ is a small positive number, e.g. 10^{-3}, and $r=|\omega|$. The elements of ω point to the positions of the nonzero taps in $\widehat{\boldsymbol{h}}(k-1)$. For example, if $r=2$, $\big|\widehat{h}_0(k-1)\big|>\xi$ and $\big|\widehat{h}_3(k-1)\big|>\xi$, then $\omega(1)=0$ and $\omega(2)=3$. Clearly we can decompose $\boldsymbol{H}(k)=\sum_{i=1}^{r}\boldsymbol{H}_i(k)$, where $\boldsymbol{H}_i(k)$ has all zero elements except $\big|\widehat{h}_{\omega(i)}(k-1)\big|/\rho$ at the diagonal position $(\omega(i)+1,\omega(i)+1)$, which matches a corresponding diagonal value in $\boldsymbol{H}(k)$ that is significantly larger than zero.

Similarly, denoting $\boldsymbol{Q}_i^{[inv]}(k)=\boldsymbol{Q}_i^{-1}(k)$ with $\boldsymbol{Q}_1^{[inv]}(k)=\boldsymbol{Q}^{[inv]}(k)$ and applying Theorem 1 yield

$$\begin{aligned}\boldsymbol{Q}_{i+1}^{[inv]}(k)&=\boldsymbol{Q}_i^{[inv]}(k)-\frac{1}{1+g_i(k)}\boldsymbol{Q}_i^{[inv]}(k)\boldsymbol{H}_i(k)\boldsymbol{Q}_i^{[inv]}(k)\\ &=\boldsymbol{Q}_i^{[inv]}(k)-\frac{\big|\widehat{h}_i(k-1)\big|}{\rho+\big|\widehat{h}_i(k-1)\big|\widetilde{q}_i(k)}\widetilde{\boldsymbol{q}}_i(k)\widetilde{\boldsymbol{q}}_i^{\mathrm{H}}(k),\end{aligned} \tag{24}$$

for $i=1,2,\cdots,r$, in which $g_i(k)=\mathrm{tr}\big\{\boldsymbol{H}_i(k)\boldsymbol{Q}_i^{[inv]}(k)\big\}$, $\widetilde{q}_i(k)$ is the $(\omega(i)+1)$th diagonal element of $\boldsymbol{Q}_i^{[inv]}(k)$, and $\widetilde{\boldsymbol{q}}_i(k)$ is the $(\omega(i)+1)$th column of $\boldsymbol{Q}_i^{[inv]}(k)$. Note that we have $\boldsymbol{S}(k)=\boldsymbol{Q}_{r+1}^{[inv]}(k)$.

We summarize the proposed ZA-RLS-II algorithm in Algorithm 2.

C. Convergence analysis

The exponential convergence of the standard RLS with exponential forgetting factor was studied in [26], which focuses on the "homogeneous" case that if $y(k)$ is exactly given as $\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{h}$, then $\widehat{\boldsymbol{h}}(k)$ converges to $\boldsymbol{h}$ exponentially fast, provided that $\boldsymbol{x}(k)$ is persistently exciting. Similarly, for the proposed algorithm, if we define a parameter estimation error vector

$$\widetilde{\boldsymbol{h}}(k)=\boldsymbol{h}-\widehat{\boldsymbol{h}}(k) \tag{25}$$

and consider the "homogeneous" case for (13) with $y(k)=\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{h}$, then

$$\begin{aligned}\widetilde{\boldsymbol{h}}(k)=&\Big(\boldsymbol{I}-\rho\boldsymbol{P}(k)\big(\boldsymbol{D}(k)-\lambda\boldsymbol{D}(k-1)\big)-\boldsymbol{P}(k)\boldsymbol{x}^*(k)\boldsymbol{x}^{\mathrm{T}}(k)\Big)\\ &\times\widetilde{\boldsymbol{h}}(k-1)+\rho\boldsymbol{P}(k)\big(\boldsymbol{D}(k)-\lambda\boldsymbol{D}(k-1)\big)\boldsymbol{h}.\end{aligned} \tag{26}$$

We are ready to provide the exponential convergence of the proposed ZA-RLS algorithms.

Theorem 2: If $\boldsymbol{P}(k)$ is invertible, then the ZA-RLS algorithms are exponentially stable.

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

Algorithm 2 ZA-RLS-II algorithm.

1: Initialize $\widehat{\boldsymbol{h}}(0)$ as a zero or small random vector. Set $\boldsymbol{Q}(0)=\frac{1}{\delta}\boldsymbol{I}$ with δ being a very small positive number, and set $\boldsymbol{Q}^{[inv]}(0)=\delta\boldsymbol{I}$. Initialize both $\boldsymbol{D}(0)$ and $\boldsymbol{D}(1)$ as zero matrix.
2: **for** time step $k=1,2,\cdots$, **do**
3: Calculate

$$\left\{\begin{array}{rcl} e(k) &=& y(k)-\boldsymbol{x}^{\mathrm{T}}(k)\widehat{\boldsymbol{h}}(k-1)\\ \boldsymbol{H}(k) &=& \mathrm{diag}\left\{\big(\big|\widehat{h}_0(k-1)\big|+\epsilon\big)/\rho,\cdots,\big(\big|\widehat{h}_L(k-1)\big|+\epsilon\big)/\rho\right\}\\ \boldsymbol{D}(k) &=& \mathrm{diag}\left\{\dfrac{1}{\big|\widehat{h}_0(k-1)\big|+\epsilon},\cdots,\dfrac{1}{\big|\widehat{h}_L(k-1)\big|+\epsilon}\right\},\ \text{for}\ \ k>1\\ \boldsymbol{Q}^{[inv]}(k) &=& \lambda\boldsymbol{Q}^{[inv]}(k-1)+\boldsymbol{x}^{*}(k)\boldsymbol{x}^{\mathrm{T}}(k)\\ \boldsymbol{Q}(k) &=& \frac{1}{\lambda}\left(\boldsymbol{Q}(k-1)-\dfrac{\boldsymbol{Q}(k-1)\boldsymbol{x}^{*}(k)\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{Q}(k-1)}{\lambda+\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{Q}(k-1)\boldsymbol{x}^{*}(k)}\right)\end{array}\right.$$

4: Given $\boldsymbol{Q}^{[inv]}(k)$, $\widehat{\boldsymbol{h}}(k-1)$, ξ, find ω according to (23), set $r=|\omega|$ and set $\boldsymbol{Q}_1^{[inv]}(k)=\boldsymbol{Q}^{[inv]}(k)$.
5: **for** $i=1,2,\cdots,r$, **do**
6: Calculate

$$\boldsymbol{Q}_{i+1}^{[inv]}(k)=\boldsymbol{Q}_{i}^{[inv]}(k)-\frac{\big|\widehat{h}_i(k-1)\big|}{\rho+\big|\widehat{h}_i(k-1)\big|\widetilde{q}_i(k)}\widetilde{\boldsymbol{q}}_i(k)\widetilde{\boldsymbol{q}}_i^{\mathrm{H}}(k)$$

7: **end for**
8: $\boldsymbol{S}(k)=\boldsymbol{Q}_{r+1}^{[inv]}(k)$.
9: Calculate

$$\left\{\begin{array}{rcl}\boldsymbol{P}(k) &=& \boldsymbol{H}(k)-\boldsymbol{H}(k)\boldsymbol{S}(k)\boldsymbol{H}(k)\\ \widehat{\boldsymbol{h}}(k) &=& \widehat{\boldsymbol{h}}(k-1)-\rho\boldsymbol{P}(k)\big(\boldsymbol{D}(k)-\lambda\boldsymbol{D}(k-1)\big)\widehat{\boldsymbol{h}}(k-1)+\boldsymbol{P}(k)\boldsymbol{x}^{*}(k)e(k)\end{array}\right.$$

10: **end for**

Proof: We choose a Lyapunov function as

$$\mathcal{V}(k)=\Big(\boldsymbol{P}^{-1}(k)\widetilde{\boldsymbol{h}}(k)-\rho\boldsymbol{D}(k)\boldsymbol{h}\Big)^{\mathrm{H}}\Big(\boldsymbol{P}^{-1}(k)\widetilde{\boldsymbol{h}}(k)-\rho\boldsymbol{D}(k)\boldsymbol{h}\Big)>0. \tag{27}$$

Applying (10) to (26) yields

$$\begin{aligned}\widetilde{\boldsymbol{h}}(k)=&\lambda\boldsymbol{P}(k)\boldsymbol{P}^{-1}(k-1)\widetilde{\boldsymbol{h}}(k-1)\\ &+\rho\boldsymbol{P}(k)\big(\boldsymbol{D}(k)-\lambda\boldsymbol{D}(k-1)\big)\boldsymbol{h}.\end{aligned} \tag{28}$$

Substituting (28) into the Lyapunov function (27) results in

$$\mathcal{V}(k)-\mathcal{V}(k-1)=\big(\lambda^2-1\big)\mathcal{V}(k-1)<0 \tag{29}$$

and

$$\mathcal{V}(k)<\lambda\mathcal{V}(k-1)<\cdots<\lambda^k\mathcal{V}(0). \tag{30}$$

This proves the exponential convergence of the ZA-RLS algorithms. Finally we conclude from $\boldsymbol{P}^{-1}(k)\widetilde{\boldsymbol{h}}(k)-\rho\boldsymbol{D}(k)\boldsymbol{h}\to\boldsymbol{0}$ that

$$\begin{aligned}\widehat{\boldsymbol{h}}(k)\to&\big(\boldsymbol{I}-\rho\boldsymbol{P}(k)\boldsymbol{D}(k)\big)\boldsymbol{h}=\\ &\Big(\boldsymbol{X}^{\mathrm{H}}(k)\boldsymbol{\Lambda}(k)\boldsymbol{X}(k)+\rho\boldsymbol{D}(k)\Big)^{-1}\boldsymbol{X}^{\mathrm{H}}(k)\boldsymbol{\Lambda}(k)\boldsymbol{X}(k)\boldsymbol{h}\end{aligned} \tag{31}$$

exponentially fast. ■

Remark 1: The rank-1 updates are a standard way of building the inversion of the covariance matrix, e.g. used in deriving RLS algorithms. The process can be unstable for ill conditioned data sets. However, in the proposed ZA-RLS algorithms, $\boldsymbol{P}(k)$ is always invertible due to regularization. Specifically, observe that the rank-1 updates (24) are guaranteed to be well conditioned owing to the regularization $\rho>0$. In fact, the regularization introduced ensures that our ZA-RLS algorithms have even better numerical stability than the standard RLS algorithm. The standard RLS algorithm is of course well known to be numerically stable under typical floating-point implementation which has sufficient precision.

Remark 2: From (31) clearly $\widehat{\boldsymbol{h}}(k)$ is a biased estimator of $\boldsymbol{h}$ for nonzero ρ. However, if $\rho=0$, the parameter estimate will have a large variance due to ill condition of the covariance matrix, especially for sparse channels. In statistical estimation theory, this is well known as bias and variance trade off in choosing the regularization parameter ρ. Note that the regularization parameter selection criteria for LMS algorithm [22], [23] are not applicable to RLS based algorithms which have much better performance that LMS based algorithms. In this paper we empirically choose appropriate values of ρ with respect to the system's signal to noise ratio (SNR) conditions. It is highly desired to investigate how to choose ρ optimally by minimizing the estimation error $\big\|\widetilde{\boldsymbol{h}}(k)\big\|^2$, defined as the squared norm of $\widetilde{\boldsymbol{h}}(k)$, using for example approximate Bayesian models. This will be our future study.

D. Computational complexity analysis

The key difference between ZA-RLS-I and ZA-RLS-II algorithms lies in how to calculate $\boldsymbol{P}(k)$. Each recursive step of the ZA-RLS-I algorithm has a computational cost on the order of $\mathsf{O}\big((L+1)^3\big)$, but for the ZA-RLS-II algorithm this is reduced to on the order of $\mathsf{O}\big(r(L+1)^2\big)$, where r is the

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

estimated sparsity level at each recursion. This is because the computation in Algorithm 2 is essentially a series of r rank-1 updates each with the complexity on the order of $\mathsf{O}\big((L+1)^2\big)$. Moreover, r varies during the recursive updating procedure. However, since $r \ll L+1$ all the time, the complexity of the ZA-RLS-II algorithm is approximately on the order of $\mathsf{O}\big((L+1)^2\big)$. This is because the underlying system that we consider is very sparse, with the true sparsity level much smaller than $L+1$. Based on the standard initialization of $\widehat{\boldsymbol{h}}(0)$, i.e., setting the elements of $\widehat{\boldsymbol{h}}(0)$ to zero or randomly to small values, which is the most widely adopted initialization for recursive identification, it is most unlikely that r at any recursion step will ever reach a high value closed to $L+1$. Rather, it is most likely that r will remain to be much smaller than $L+1$ all the time. In the simulation study, we will verify this analysis.

Thus the computational cost of the ZA-RLS-II algorithm is only slightly higher than that of the standard RLS algorithm, which also has the complexity on the order of $\mathsf{O}\big((L+1)^2\big)$. More specifically, at each recursive step, in addition to update/store the matrix variable $\boldsymbol{Q}(k)$ as also required by the conventional RLS algorithm, the ZA-RLS-II algorithm is requires to update/store $\boldsymbol{Q}^{[inv]}(k)$ but this only involves a negligible additional computational cost.

It is also obvious that the proposed ZA-RLS-II algorithm, the l_1-RLS algorithm [16] and the SPARLS algorithm [14] all have similar computational complexity, on the order of $\mathsf{O}\big((L+1)^2\big)$ at each recursive step.

IV. Simulation Study

Simulations were carried out with the channel input signal $x(k)$ taking values from the $M=64$ quadrature amplitude modulation (QAM) symbol set. The transmit signal was normalized to have a unity average symbol energy and thus the average energy per bit was given by $E_b = \frac{1}{\log_2(M)} = \frac{1}{6}$. The received SNR was defined as $10\log_{10}(E_b/N_o)$. The sparse multipath channel had a length of 100, i.e. $L=99$, with its true sparsity level denoted by $r_{\rm true}$, which was the number of the nonzero elements in $\boldsymbol{h}$. Based on the average results over 100 random trials, we tested the performance of the proposed algorithms in three aspects: (i) the performance comparison with some other well-known l_1-norm based adaptive algorithms; (ii) the effects of the sparsity levels on the proposed sparse algorithms; and (iii) the effects of the regularization parameter with respective to the SNR. For each trial, the training data length was set to 1000. The positions of the significant $r_{\rm true}$ taps were randomly selected within the channel length. The CIR $\boldsymbol{h}$ was kept constant in the first half of the each trial, and it suddenly changed at the beginning of the second half of the each trial in terms of both the tap positions and values. The values of channel taps followed the Gaussian distribution and were normalized to $\|\boldsymbol{h}\|^2=1$. The estimation performance was evaluated based on the average mean absolute deviation (MAD), defined as

$$\mathrm{MAD}\big\{\widehat{\boldsymbol{h}}(k)\big\} = \mathrm{E}\big\{\big|\widehat{\boldsymbol{h}}(k)-\boldsymbol{h}\big|\big\} = \mathrm{E}\Big\{\sum_{i=0}^{L}\big|\widehat{h}_i(k)-h_i\big|\Big\}, \tag{32}$$

and calculated based on the average value over 100 independent random trials.

Comparison with other l_1-norm based complex-valued sparse adaptive algorithms: Firstly we compared the proposed algorithms with several known l_1-norm based algorithms under the conditions of SNR $=15$ dB and 30 dB, respectively, while the number of significant taps was fixed to $r_{\rm true}=10$. Specifically, the proposed ZA-RLS-I and ZA-RLS-II algorithms were compared with the ZA-LMS, the ZA-NLMS, the conventional RLS, the SPARLS algorithm [14], the l_1-RLS algorithm [16] and the oracle-RLS algorithm which is just the ordinary RLS given the nonzero tap locations. The oracle-RLS algorithm obviously will attain the best performance but it is impractical as the nonzero tap locations are unknown. The reasons that we chose the SPARLS algorithm [14] and the l_1-RLS algorithm [16] as baseline algorithms are because they are both based on the l_1 cost function with different algorithm designs, and

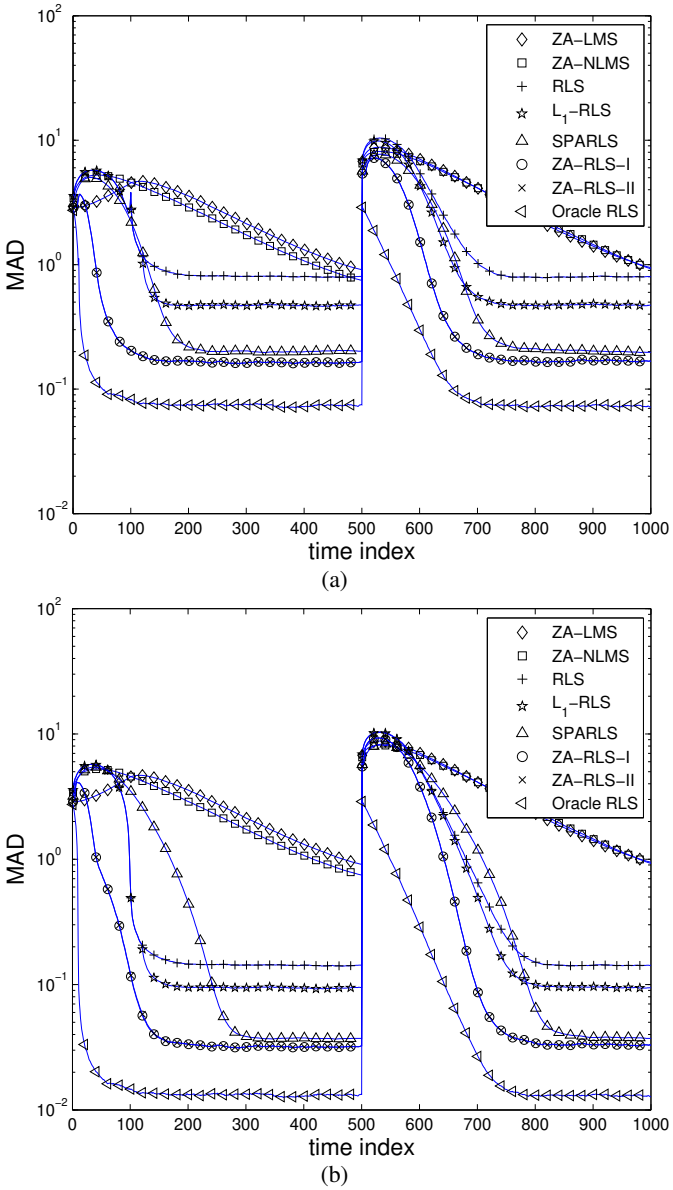

Fig. 1. Comparison of the MADs of the parameter estimates for various adaptive algorithms: (a) SNR $=15$ dB and (b) SNR $=30$ dB. The channel input signal is complex-valued and the channel sparsity level $r_{\rm true}=10$.

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

they have the same computational complexity as the proposed ZA-RLS-II. Moreover, all these algorithms are designed for complex-valued channel identification.

For the ZA-LMS algorithm, we set $\mu = 0.01$, and $\rho_{ZA-LMS} = 0.01$, while for the ZA-NLMS algorithm, we set $\mu = 0.8$, and $\rho_{ZA-LMS} = 0.01$, since these values were found to give the best results for the respective algorithms. For all the RLS based algorithms, the forgetting factor $\lambda = 0.975$. We also set $\xi = 0.0001$ in the ZA-RLS-II. The parameters used in the SPARLS algorithm [14] and the l_1-RLS algorithm [16] were tuned empirically to give the best performance possible. In addition, in the case of the l_1-RLS algorithm, the performance was very bad at the beginning of the data set, so the ordinary RLS algorithm was used for the first 120 data points. For the ZA-RLS-I and ZA-RLS-II algorithms, we set $\rho = 0.5$ for SNR $= 15$ dB and $\rho = 0.1$ for SNR $= 30$ dB.

The MAD results obtained by various adaptive algorithms are compared in Fig. 1 (a) and (b), respectively, for the two SNR settings. As expected, all the RLS based algorithms attain much better MAD performance than the ZA-LMS and ZA-NLMS algorithms. It is also seen that the results of the two proposed algorithms are identical, and they significantly outperform the l_1-RLS algorithm. Fig. 1 also shows that the ZA-RLS-II algorithm achieves smaller steady state error with faster convergence rate, compared to the SPARLS algorithm. These results are significant, particularly considering that the ZA-RLS-II algorithm has similar computational complexity as the l_1-RLS algorithm and the SPARLS algorithm. In Fig. 2, we plot the estimated sparsity level r as recoded by the ZA-RLS-II algorithm at each recursive step, average over the 100 independent random trials. It can be observed from Fig. 2 that $r \ll L + 1$ all the time.

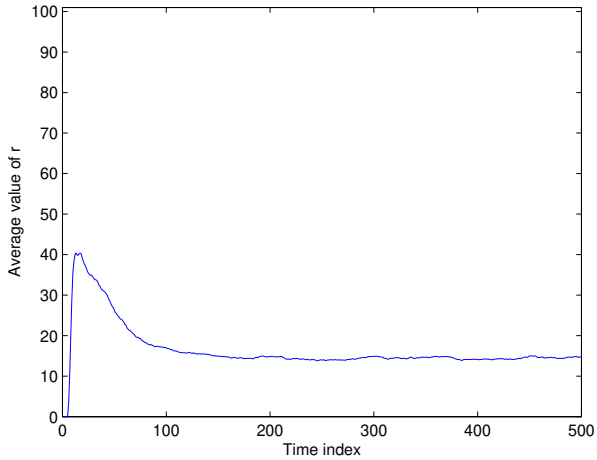

Fig. 2. Estimated sparsity level r as recorded by the ZA-RLS-II algorithm at each recursive step, averaged over 100 runs. The channel input signal is complex-valued, the channel length $L + 1 = 100$, the channel sparsity level $r_{\text{true}} = 10$, and SNR $= 15$ dB.

Performance evaluation for various sparsity levels: In order to investigate the performance of the proposed algorithms for different sparsity levels, we further experimented by changing the number of significant taps r_{true}, also under the conditions of SNR $= 15$ dB and 30 dB. Specifically, $r_{\text{true}} = 1$, 5, 20 and 50 was experimented. For clarity only the oracle-RLS and ordinary RLS algorithms were used for comparison, and

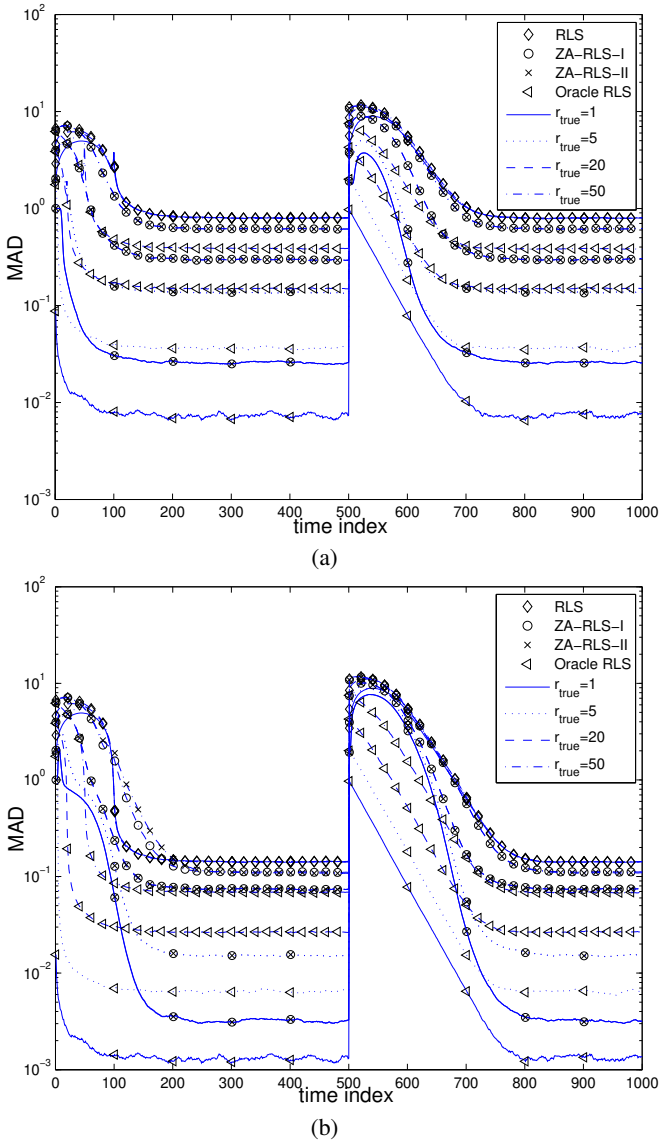

Fig. 3. Comparison of the MADs of the parameter estimates for various sparsity levels: (a) SNR $= 15$ dB and (b) SNR $= 30$ dB. The channel input signal is complex-valued.

the results obtained were plotted in Fig. 3 (a) and (b) for the two SNR settings, respectively. Since the channel taps were normalized in spite of different sparsity levels, we used different values of ρ as appropriate for the ZA-RLS-I and ZA-RLS-II. Specifically, when SNR $= 30$ dB, we empirically set $\rho = 0.1$ for $r_{\text{true}} = 1$, 5 and 20, but $\rho = 0.01$ for $r_{\text{true}} = 50$. When SNR $= 15$ dB, we empirically found $\rho = 1$ for $r_{\text{true}} = 1$ and 5, $\rho = 0.3$ for $r_{\text{true}} = 20$, while $\rho = 0.05$ for $r_{\text{true}} = 50$. The results of Fig. 3 clearly show that it is most beneficial to use the sparse adaptive algorithms when the sparsity level is high, i.e. the value of r_{true} is small. As r_{true} increases to the full channel length, the oracle-RLS becomes the ordinary RLS algorithm. In this case, there exists no sparsity to be exploited and ρ should be chosen as very small positive number close to zero just for numerical stability consideration.

Performance evaluation for different regularization parameter and SNR settings: To study the effects of ρ with respect to SNR levels for a fixed sparsity level of $r_{\text{true}} = 10$, we recorded the results of MAD$\{\widehat{\boldsymbol{h}}(300)\}$ as the functions of

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

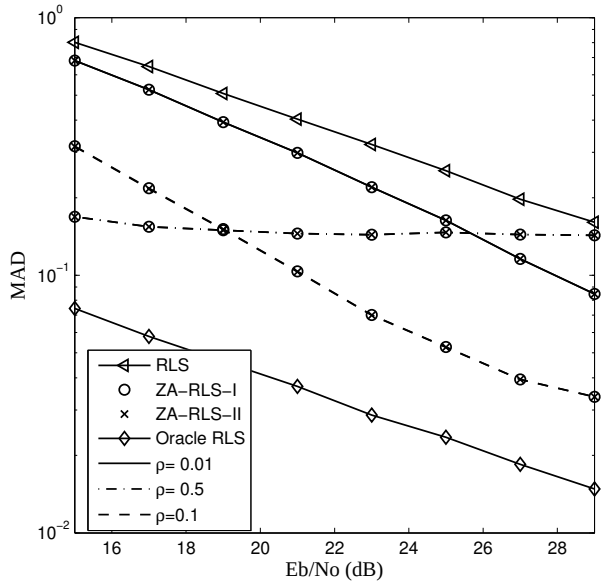

Fig. 4. Comparison of the steady-state mean absolute deviations of the parameter estimates as functions of SNR. The channel input signal is complex-valued, the channel sparsity level $r_{\text{true}} = 10$, and three different regularization parameter values are tested for the proposed ZA-RLS algorithms.

SNR in Fig. 4, based on $\rho = 0.5$, $\rho = 0.1$ and $\rho = 0.01$, respectively. It can be clearly seen from Fig. 4 that for the proposed ZA-RLS algorithms, a relatively larger ρ should be used under high noise condition, while when the noise level is low, ρ should be set relatively small. However, too small ρ leads to poor performance.

Comparison with the l_1-norm based real-valued sparse adaptive algorithms of [19]: The l_1-norm based sparse RLS algorithms of [19] are designed for real-valued signals, and they cannot be directly applied to our application of sparse channel identification involving complex-valued signals. By contrast, our algorithm and the other algorithms compared in the above experiment can deal with both real-valued and complex-valued signals. In order to compare our algorithm with the algorithms of [19], we specifically design an application involving real-valued signals. The experiment is the same as the experiment of Fig. 1, but the input signal $x(k)$ is changed to be real-valued and is generated as a uniformly and randomly distributed signal in $[0,\ 1]$. The variance of the noise is set to 0.05, and the resulting SNR is approximately 21 dB. Note that the two online algorithms mentioned in [19], the OCCD-TWL and OCCD-TNWL, only the OCCD-TWL is explicitly derived in [19]. Since the paper [19] does not provide how the OCCD-TNWL is actually realized, we implement the OCCD-TNWL based on our understanding from the offline TNWL algorithm given in [19]. The OCCD-TNWL algorithm is much more complicated than the OCCD-TWL. Specifically, at each recursion, the key adaptive parameter of the algorithm is weighted and the weighting factor depends on the full RLS channel estimate. Therefore, an additional full RLS algorithm is required to run in parallel in order to provide the full channel estimate at each recursion.

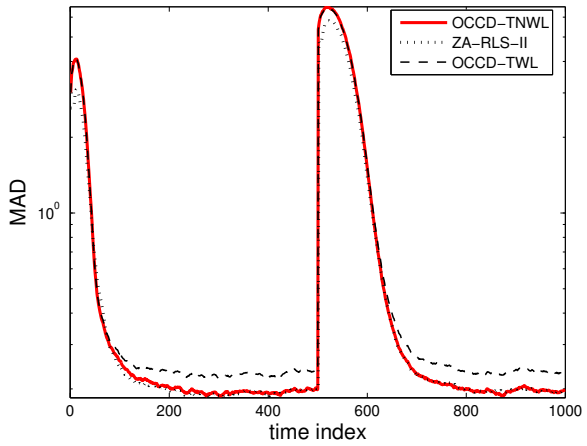

Fig. 5. Comparison of the MADs of the parameter estimates for various adaptive algorithms. The channel input signal is real-valued, the channel sparsity level $r_{\text{true}} = 10$ and SNR = 21 dB.

The MAD performance of our ZA-RLS-II, OCCD-TWL and OCCD-TNWL are compared in Fig. 5, where it can be seen that both the ZA-RLS-II and OCCD-TNWL outperform the OCCD-TWL. It can also be seen from Fig. 5 that our ZA-RLS-II and the OCCD-TNWL achieve the same steady-state performance but our ZA-RLS-II has an additional advantage of slightly better initial transition performance.

V. Conclusions

In this contribution, we have introduced two ZA-RLS algorithms for the sparse channel identification problem by using the l_1-norm sparsity constraint adaptively. The basic idea in achieving a closed-form solution is to use an adaptively weighted l_2-norm of the parameter vector term to approximate the l_1-norm of the parameter vector, in which the weighting factors are readily calculated as the inversion of the associated l_1-norm of the parameter estimates. As a variant of the ZA-RLS-I, the ZA-RLS-II algorithm has focused on improving the computational efficiency by exploiting the channel sparsity and matrix theory. Consequently, the computational complexity of the ZA-RLS-II algorithm is only slightly higher than the standard RLS algorithm. The proposed ZA-RLS-II algorithm is compared with a number of adaptive algorithms which also use l_1-norm sparsity constraints, and the simulations results have demonstrated that the proposed ZA-RLS approach is highly effective in real-time sparse channel estimation. In particular, it has been shown that the proposed ZA-RLS-II algorithm outperforms the existing l_1-RLS and SPARLS algorithms which have similar computational complexity and also use the same l_1 cost function but with different approximations for algorithm design. Our future work will study efficient tuning algorithm for optimizing the regularization parameter.

Acknowledgements

The authors would like to thank the reviewers for their constructive comments that helped to improve the paper. The authors are also thankful to Behtash Babadi for providing the SPARLS Matlab code.

References

[1] S. Haykins, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2016.2533664, IEEE Transactions on Vehicular Technology

[2] D. L. Donoho and M. Elad, "Optimally sparse representation in general (non-orthogonal) dictionaries via l_1 minimization," *Proc. Nat. Acad. Sci. USA*, vol. 100, no. 5, pp. 2197–2202, Mar. 2003.

[3] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Information Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[4] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Information Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[5] D. Needell, J. A. Tropp, and R. Vershynin, "Greedy signal recovery review," in *Proc. 42nd Asilomar Conf. Signals, Systems, and Computers* (Pacific Grove, CA), Oct. 26-29, 2008, pp. 1048–1050.

[6] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Applied Computational Harmonic Analysis*, vol. 53, no. 12, pp. 301–321, May 2009.

[7] T. Blumensath and M. E. Davies, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE J. Selected Topics Signal Processing*, vol. 4, no. 2, pp. 298–309, Apr. 2010.

[8] S. Foucart, "Hard thresholding pursuit: An algorithm for compressive sensing," *SIAM J. Numerical Analysis*, vol. 49, no. 6, pp. 2543–2563, 2011.

[9] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing: I. motivation and construction," in *Proc. 2010 IEEE Information Theory Workshop* (Cairo Egypt), Jan. 6-8, 2010, pp. 1–5.

[10] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for system identification," in *Proc. ICASSP 2009* (Taipei, China), Apr. 19-24, 2009, pp. 3125–3128.

[11] O. Taheri and S. A. Vorobyov, "Sparse channel estimation with l_p-norm and reweighed l_1-norm penalized least mean squares," in *Proc. ICASSP 2011* (Prague, Czech Republic), May 22-27, 2011, pp. 2864–2867.

[12] G. Gui and F. Adachi, "Improved least mean square algorithm with application to adaptive sparse channel estimation," *EURASIP J. Wireless Communications and Networking*, vol. 204, pp. 1–18, 2013.

[13] Y. T. Gu, J. Jin, and S. Mei, "l_0-norm constraint LMS algorithm algorithm for sparse system identification," *IEEE Signal Processing Letters*, vol. 16, no. 9, pp.774–777, Sep. 2009.

[14] B. Babadi, N. Kalouptsidis, and V. Tarokh, "SPARLS: The sparse RLS algorithm," *IEEE Trans. Signal Processing*, vol. 58, no. 8, pp. 4013–4025, Aug. 2010.

[15] B. Dumitrescu, A. Onose, P. Helin, and I. Tabus, "Greedy sparse RLS," *IEEE Trans. Signal Processing*, vol. 60, no. 5, pp. 2194–2207, May 2012.

[16] E. M. Eksioglu and A. Korhan, "RLS algorithm with convex regularization," *IEEE Signal Processing Letters*, vol. 18, no. 8, pp.470–473, Aug. 2011.

[17] E. M. Eksioglu, "Sparsity regularised recursive least squares adaptive filtering," *IET Signal Processing*, vol. 5, no. 5, pp. 480–487, Aug. 2011.

[18] Y. Zakharov and V. H. Nascimento, "Sparse RLS adaptive filter with diagonal loading," in *Proc. 46th Asilomar Conf. Signals, Systems and Computers* (Pacific Grove, CA), Nov. 4-7, 2012, pp. 806–810.

[19] D. Angelosante, J. A. Bazerque, and G. B. Giannakis, "Online adaptive estimation of sparse signals: Where RLS meets the l_1-Norm," *IEEE Trans. Signal Processing*, vol. 58, no. 7, pp. 3436–3447, Jul. 2010.

[20] Z. Liu, Y. Liu, and C. Li, "Distributed sparse recursive least-squares over networks," *IEEE Trans. Signal Processing*, vol. 62, no. 6, pp. 1386–1395, Mar. 2014.

[21] Y. V. Zakharov and V. H. Nascimento, "DCD-RLS adaptive filters with penalties for sparse identification," *IEEE Trans. Signal Processing*, vol. 61, no. 12, pp. 3198–3213, Jun. 2013.

[22] K. Shi and P. Shi, "Convergence analysis of sparse LMS algorithms with l_1-norm penalty based on white input signal," *Signal Processing*, vol. 90, no. 12, pp 3289–3293, Dec. 2010.

[23] Y. Liu, C. Li, and Z. Zhang, "Diffusion sparse least-mean squares over networks," *IEEE Trans. Signal Processing*, vol. 90, no. 8, pp 4480–4485, Aug. 2012.

[24] M. A. T. Figueiredo and R. D. Nowak, "An EM algorithm for wavelet-based image restoration," *IEEE Trans. Image Processing*, vol. 12, no. 8, pp. 906–916, Aug. 2003.

[25] K. S. Miller, "On the inverse of the sum of matrices," *Mathematics Magazine*, vol. 54, no. 2, pp. 67–72, 1981.

[26] R. M. Johnstone, C. R. Johnson, R. R. Bitmead, and B. D. O. Anderson, "Exponential convergence of recursive least squares with exponential forgetting factor," in *Proc. 21st IEEE Conf. Decision and Control* (Orlando, FL), Dec.8-10, 1982, pp. 994–997.

Xia Hong (SM'02) received her university education at National University of Defense Technology, P. R. China (BSc, 1984, MSc, 1987), and University of Sheffield, UK (PhD,1998), all in automatic control. She worked as a research assistant in Beijing Institute of Systems Engineering, Beijing, China from 1987-1993. She worked as a research fellow in the Department of Electronics and Computer Science at University of Southampton from 1997-2001. She is currently a Professor at Department of Computer Science, School of Mathematical and Physical Sciences, University of Reading. She is actively engaged in research into nonlinear systems identification, data modeling, estimation and intelligent control, neural networks, pattern recognition, learning theory and their applications. She has published over 200 research papers, and coauthored a research book. Professor Hong was awarded a Donald Julius Groen Prize by IMechE in 1999.

Junbin Gao graduated from Huazhong University of Science and Technology (HUST), China in 1982 with BSc degree in Computational Mathematics and obtained PhD from Dalian University of Technology, China in 1991.

He is a Professor of Big Data Analytics in the University of Sydney Business School at the University of Sydney, and was a Professor in Computer Science in the School of Computing and Mathematics at Charles Sturt University, Australia before 2016. He was a lecturer, a senior lecturer in Computer Science from 2001 to 2005 at University of New England, Australia. From 1982 to 2001 he was an associate lecturer, lecturer, associate professor and professor in Department of Mathematics at HUST.

His main research interests include machine learning, data analytics, Bayesian learning and inference, and image analysis.

Sheng Chen (M'90-SM'97-F'08) received his BEng degree from the East China Petroleum Institute, Dongying, China, in 1982, and his PhD degree from the City University, London, in 1986, both in control engineering. In 2005, he was awarded the higher doctoral degree, Doctor of Sciences (DSc), from the University of Southampton, Southampton, UK.

From 1986 to 1999, He held research and academic appointments at the Universities of Sheffield, Edinburgh and Portsmouth, all in UK. Since 1999, he has been with Electronics and Computer Science, the University of Southampton, UK, where he currently holds the post of Professor in Intelligent Systems and Signal Processing. Dr Chen's research interests include adaptive signal processing, wireless communications, modeling and identification of nonlinear systems, neural network and machine learning, intelligent control system design, evolutionary computation methods and optimization. He has published over 550 research papers.

Dr. Chen is a Fellow of IET, a Distinguished Adjunct Professor at King Abdulaziz University, Jeddah, Saudi Arabia, and an ISI highly cited researcher in engineering (March 2004). He was elected to a Fellow of the United Kingdom Royal Academy of Engineering in 2014.

0018-9545 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.