Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2012, Article ID 452806, 12 pages
doi:10.1155/2012/452806

Research Article

CP-Based SBHT-RLS Algorithms for Tracking Channel Estimates in Multicarrier Modulation Systems

H. Ali

School of Electrical Engineering and Computer Science, Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan NSW2308, Australia

Correspondence should be addressed to H. Ali, hassan.ali@newcastle.edu.au

Received 18 July 2011; Accepted 3 October 2011

Academic Editor: Yin-Tsung Hwang

Cyclic prefix (CP) in multicarrier modulation systems has been considered as an alternative to the training sequences to track channel estimates. In this paper, two new algorithms are developed that exploit CP from their data detection part and employ systolic block Householder transformation recursive least squares (SBHT-RLS) algorithms for channel tracking in multicarrier systems. The new methods are compared with the existing CP exploiting correlation matrix based block RLS (CMB-RLS) channel tracking approach to outline their relative advantages. Aspects of computational complexity and parallel implementation are addressed, and the algorithms are tested in terms of their channel estimation and tracking capabilities. Performance of the algorithms is also evaluated for varying forgetting factor parameter values, constellation size, and word lengths. Floating-point and fixed-point simulations are tailored to illustrate pertinent tradeoffs.

1. Introduction

Over the last two decades, multicarrier modulation has received considerable interest for its use in wireless and wireline communication systems [1–4]. It has been adopted in many communication standards, including digital audio broadcasting (DAB) [5], digital video broadcasting (DVB) [6], high-speed modems over digital subscriber lines (xDSLs) [4], and local area mobile wireless broadband [7].

Most multicarrier systems use coherent detection of data symbols, which requires reliable estimation of channel at the receiver. Channel state information is also necessary for techniques such as channel shortening [8], adaptive modulation/loading, and/or power control [9]. In applications such as discrete multitone (DMT) xDSL [4], channel is estimated through some initial training process, and retraining is required to track the channel variation. To avoid the system overhead due to retraining and thus to track the channel more efficiently, in [10], a correlation matrix based block recursive least-squares (CMB-RLS) algorithm is proposed. The algorithm takes advantage of the inherent redundancy introduced by the cyclic prefix (CP) to blindly estimate the channel. In [11], performance of the algorithm is analyzed considering both the effect of channel noise and decision error. The algorithm is further explored in [12], where its performance is analyzed considering the impact of exponential forgetting factor values, constellation size, and channel nulls. Also, in [13], the method is used in single-carrier (SC) modulation with frequency domain equalization (FDE) to maintain both system performance and throughput.

While CP-based CMB-RLS approach is standard complaint, there are two basic problems that make it unsuitable for real-time implementation. First, it relies on computation of inverse of the correlation matrix $\overline{\mathbf{\Phi}}$ per time update. The computational cost of performing the required matrix inversion in real time can be prohibitively high for a system with a large channel length (To reduce the computational complexity and thus processing power, this inversion cannot be done recursively using *Matrix inversion Lemma* (such as in conventional RLS (CRLS) algorithm [14]).). Second, the direct inversion and recursive inversion approaches are known to severely limit parallelism and pipelining that can effectively be applied in the practical implementation.

Usually, to minimize the round off error, matrix inversions are done with general-purpose digital signal processing (DSP) devices/processors using floating-point arithmetic. A disadvantage of this approach, however, is severe processing

power limitation due to small number of floating-point processing units commonly available per device. Specialized hardware with high-processing power is therefore required to execute requisite computations in real time. An appealing alternative for implementation is not to do this inversion explicitly and solve the problem through a computationally cheaper approach that works directly with data matrix and is realizable on the systolic array architecture offering large amounts of parallelism for high-speed very large scale integration (VLSI) implementation. In VLSI implementation, floating-point arithmetic units are more complex than those of fixed-point arithmetic, involving extra hardware overhead and more clock cycles [15]. Hence, the bit-level systolic architecture must be implemented with fixed-point arithmetic.

The QR decomposition (QRD) approaches for RLS problem have played an important role in adaptive signal processing, adaptive equalization, and adaptive spectrum estimation [16]. It is generally agreed that QRD-RLS algorithms are one of the most promising RLS algorithms, due to their numerical stability [17, 18] and suitability for VLSI implementation [19, 20]. There are three approaches to QRD-RLS problem, namely, Givens rotation (GR), modified Gram-Schmidt (MGS), and Householder transformation (HT) method. These methods have been successfully applied to the development of the QRD-RLS systolic array [16, 21–24]. Because HT generally outperforms GR and MGS methods under finite precision computations (see the references in [16]), and in the context of our application the channel needs to be updated for each block input data matrix, we focus our attention to the QRD-RLS algorithm based on block HT. Notice that HT is a well-known rank-ν update approach and is one of the most efficient methods to compute QRD (Rank-1 updating fast QRD-RLS algorithms (where QRD is updated after the original data matrix has been modified by the addition and deletion of a row or column) [14] are not suitable here in particular due to high throughput (here the term throughput is used to indicate total number of data vectors at the input of the RLS algorithm) and speed requirements.). In [24, 25], Liu et al. investigated one such QRD-RLS algorithm using block HT. The work in [24] describes the block HT implementation on a systolic array and its application to RLS algorithm called systolic block HT-RLS (SBHT-RLS). So far, SBHT-RLS is used in beamforming and linear predication applications but has not been applied for channel tracking in high-throughput multicarrier applications. The algorithm is well known for its computational efficiency, very good numerical properties, and parallel processing implementation advantages.

In this paper, we develop two new CP exploiting SBHT-RLS approaches for adaptive channel estimation in multicarrier systems.

The first approach is based on SBHT-RLS approach of Liu et al.. In its original form, the SBHT-RLS does not provide access to channel weights, as its use has been limited to the problem seeking an estimate of output error signal. In the context of our application, the proposed approach finds the channel explicitly. In order to differentiate between the two techniques, the new method will be referred to as CP-based *Direct* SBHT-RLS approach. The proposed scheme is computationally efficient and can be mapped to triangular systolic arrays for efficient parallel implementation. Unfortunately, the scheme suffers from a major drawback, namely, back substitution, which is a costly operation to perform in array structure [26, 27].

The second approach relies on inverse factorizations to calculate least squares channel coefficients (weight vector) without back substitution. This approach also employs SBHT to recursively update the channel coefficients and thus preserves the inherent stability property of SBHT-RLS approach. The derivation of the inverse factorization method in this paper is done by generalizing the *Extended* QRD-RLS algorithm to block RLS case [28]. For this reason, this method will be referred to as CP-based *Extended* SBHT-RLS approach. We underscore here that this simple and straightforward derivation is different than the previous challenging work on block RLS using inverse factorizations in [29, 30]. Computational complexity of this scheme is equivalent to the first proposed scheme, but unlike the first scheme it is fully amenable to VLSI implementation and also results in improved steady-state performance.

For the sake of brevity, in the rest of this paper, we refer to the CP based CMB-RLS as CPE1, *Direct* SBHT-RLS as CPE2, and *Extended* SBHT-RLS as CPE3. Also, for uniformity, we closely follow the notation that appears in [10].

The paper is organized as follows. In the next section, we provide an overview of the DMT system model [10]. Section 3 explains the newly proposed algorithms, followed by a discussion on their computational complexity and systolic array implementation in Section 4. In Section 5, illustrating floating- and fixed-point simulations are conducted, while conclusions are drawn in Section 6. Some results contained in this paper have been presented/accepted for presentation in [31, 32].

Notation. $(\cdot)^T, (\cdot)^*$, and $E[\cdot]$ denote transpose, complex conjugate, and expectation operation. The Matlab notation $\mathbf{X}(:, m : m')$ is used to to denote the submatrix of $\mathbf{X}$ that contains the columns m to m'. $\mathbf{x}(n : n')$ denotes the subvector of $\mathbf{x}$ comprising of entries n through n'. $\mathbf{I}_n$ denotes identity matrix of size n, $\mathbf{0}$ denotes the all zeros matrix of appropriate dimensions, and $j = \sqrt{-1}$. The meaning of other variables will be clear from the context.

2. System Model

We consider a high-speed DMT data transmission system over digital subscriber lines, shown in Figure 1. The system has $m/2$ complex parallel subchannels and illustrates the typical CP based adaptive channel estimation task, which is our main concern in this paper. Let $\{s_n\}$ represent the data sequence to be transmitted over the channel. This input data is buffered to blocks, and each data block is divided into $m/2$ bit streams and then mapped to quadrature amplitude modulation (QAM) constellation points $X_{i,k}, i = 0, \ldots, m/2-1$ at time k. After m-point inverse fast Fourier transform (IFFT) on the kth DMT block $\mathbf{X}_k = [X_{0,k}, X_{1,k}, \ldots, X_{m-1,k}]^T$

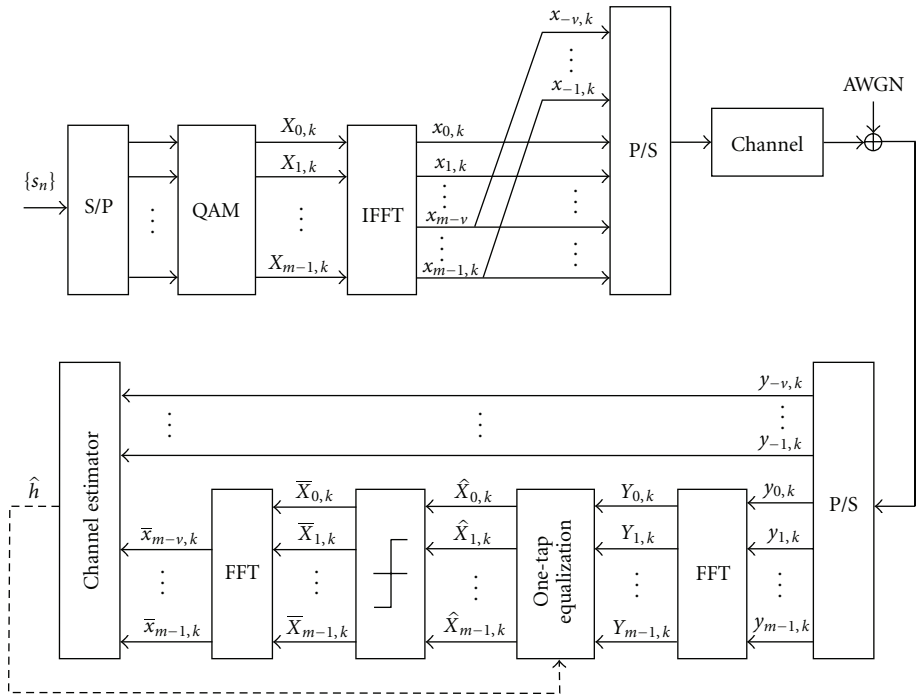

Figure 1: Multicarrier system with CP-based adaptive channel estimation.

(here the last $m/2$ samples are just the conjugates of the first $m/2$ samples), the modulated real valued time domain signal is $\mathbf{x}_k = [x_{0,k}, x_{1,k}, \ldots, x_{m-1,k}]^T$. A CP $\mathbf{x}_k^{(f)} = [\overline{x}_{m-v,k}, \ldots, \overline{x}_{m-1,k}]^T$, where $x_{-i,k} = x_{m-i,k}$ and $i = 1, \ldots, v$, is then appended in front of $\mathbf{x}_k$ before transmission through the channel $H(z) = \sum_{l=0}^{v} h_{l,k} z^{-l}$, having impulse response $\mathbf{h}_k = [h_{0,k}, h_{1,k}, \ldots, h_{v,k}]^T$ of length $r = v + 1$. At the receiver, the prefix part $\mathbf{y}_k^{(f)} = [y_{-v,k}, \ldots, y_{-1,k}]^T$ is removed.

The relationship between prefix part $\mathbf{y}_k^{(f)}$ and the transmitted signal may be expressed as [10]

$$\mathbf{y}_k^{(f)} = \mathbf{A}_k \mathbf{h}_k + \mathbf{n}_k^{(f)}, \tag{1}$$

where

$$\mathbf{A}_k = \begin{bmatrix} x_{-v,k} & x_{m-1,k-1} & \cdots & x_{m-v,k-1} \\ \ddots & \ddots & \ddots & \vdots \\ \ddots & \ddots & \ddots & \vdots \\ x_{-1,k} & \cdots & x_{-v,k} & x_{m-1,k-1} \end{bmatrix} = [\mathbf{a}_{0,k}, \mathbf{a}_{1,k}, \ldots, \mathbf{a}_{v,k}], \tag{2}$$

$\mathbf{a}_{j,k}$ is the jth column of $\mathbf{A}_k$, $\mathbf{n}_k^{(f)} = [n_{-v,k}, \ldots, n_{-1,k}]^T$, and $n_{i,k} \sim \mathcal{N}(0, \sigma^2)$ is the channel noise.

After the FFT operation on $\mathbf{y}_k = [y_{0,k}, y_{1,k}, \ldots, y_{m-1,k}]^T$, the demodulated signal is $\mathbf{Y}_k = [Y_{0,k}, Y_{1,k}, \ldots, Y_{m-1,k}]^T$. The CP removes interblock interference (IBI) between $\mathbf{X}_k$'s. The received symbols can thus be written as

$$Y_{i,k} = X_{i,k} \mathcal{H}_{i,k} + N_{i,k}, \quad i = 0, \ldots, m-1, \tag{3}$$

where $\mathcal{H}_{i,k} = \sum_{l=0}^{v} h_{l,k} e^{-j((2\pi i l)/m)}$ is the channel frequency response and $N_{i,k} = (1/\sqrt{m}) \sum_{l=0}^{m-1} n_{l,k} e^{-j((2\pi i l)/m)} \sim \mathcal{N}(0, \sigma^2)$ is the noise of the ith subchannel.

To get the estimation of $X_{i,k}$ from $Y_{i,k}$, a one-tap minimum mean square error (MMSE) equalizer $w_{i,k} = (\Gamma_i^{1/2} \mathcal{H}_{i,k}^*)/(\Gamma_i \|\mathcal{H}_{i,k}\|^2 + \sigma_i^2)$, where $i = 0, \ldots, m-1$ and $\Gamma_i = E[\|X_{i,k}\|^2]$, is then employed at the ith channel. The estimated data is then $\hat{X}_{i,k} = Y_{i,k} w_{i,k}$. The decision is then made on $\hat{X}_{i,k}$ to get the final output $\overline{X}_{i,k} = q(\hat{X}_{i,k})$, where $q(\cdot)$ is the decision operation.

3. CP-Based SBHT-RLS Algorithms

3.1. CP-Based Direct SBHT-RLS Algorithm (CPE2). Based on the CP data model (1), we define $nv \times r$ weighted data matrix and the $nv \times 1$ weighted received vector in a recursive manner as

$$\dot{\mathbf{A}}_k = \Lambda \begin{bmatrix} \mathbf{A}_{k-(n-1)} \\ \mathbf{A}_{k-n} \\ \vdots \\ \mathbf{A}_k \end{bmatrix} = \begin{bmatrix} \lambda^{1/2} \dot{\mathbf{A}}_{k-1} \\ \hline \mathbf{A}_k \end{bmatrix}, \tag{4}$$

$$\dot{\mathbf{y}}_k^{(f)} = \Lambda \begin{bmatrix} \mathbf{y}_{k-(n-1)}^{(f)} \\ \mathbf{y}_{k-n}^{(f)} \\ \vdots \\ \mathbf{y}_k^{(f)} \end{bmatrix} = \begin{bmatrix} \lambda^{1/2} \dot{\mathbf{y}}_{k-1}^{(f)} \\ \hline \mathbf{y}_k^{(f)} \end{bmatrix}, \tag{5}$$

where Λ is an $n\nu \times n\nu$ block-diagonal forgetting matrix of the form

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda^{(n-1)/2}\mathbf{I}_\nu & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \lambda^{1/2}\mathbf{I}_\nu & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I}_\nu \end{bmatrix}, \tag{6}$$

with forgetting factor across blocks $0 < \lambda \leq 1$. The forgetting factor λ is incorporated in the scheme to avoid overflow in the processors as well as to facilitate nonstationary data updating [25].

Suppose that at the $(k-1)$th update we have QRD

$$\mathbf{Q}_{k-1}\dot{\mathbf{A}}_{k-1} = \begin{bmatrix} \mathbf{R}_{k-1} \\ \hline \mathbf{0} \end{bmatrix}, \tag{7}$$

where $\mathbf{Q}_{k-1}$ is an $(n-1)\nu \times (n-1)\nu$ orthogonal matrix and $\mathbf{R}_{k-1}$ is a $r \times r$ upper triangular matrix.

Now by denoting $\overline{\mathbf{Q}}_{k-1} = \left[\begin{array}{c|c} \mathbf{Q}_{k-1} & \mathbf{0} \\ \hline \mathbf{0}^T & \mathbf{I}_\nu \end{array}\right]$, we then have

$$\overline{\mathbf{Q}}_{k-1}\dot{\mathbf{A}}_k = \begin{bmatrix} \mathbf{R}_{k-1} \\ \hline \mathbf{0} \\ \hline \mathbf{A}_k \end{bmatrix}. \tag{8}$$

A $n \times n$ HT matrix $\mathbf{T}$ is of the form $\mathbf{T} = \mathbf{I}_n - \beta\mathbf{v}\mathbf{v}^T$, where $\beta = 2/\mathbf{v}^T\mathbf{v} = 2/\|v\|^2$. When a vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ is multiplied by $\mathbf{T}$, it is reflected in the hyperplane defined by $\text{span}\{\mathbf{v}\}^\perp$. Choosing $\mathbf{v} = \mathbf{x} \pm \|\mathbf{x}\|_2\mathbf{e}_1$, where $\mathbf{e}_1 = [1, 0, 0, \ldots, 0]^T$, then $\mathbf{x}$ is reflected onto $\mathbf{e}_1$ by $\mathbf{T}$ as: $\mathbf{T}\mathbf{x} = \pm\|\mathbf{x}\|_2\mathbf{e}_1$.

A series of HTs are then used to zero out $\mathbf{A}_k$ in the right-hand side of (8). Let $\mathbf{H}_k = \mathbf{H}_k^{(r)}\mathbf{H}_k^{(r-1)} \cdots \mathbf{H}_k^{(1)}$ (a sequence of r-ordered matrix multiplications), where $\mathbf{H}_k^{(i)}$ denotes the ith HT matrix (which zeroes out ith column of updated $\mathbf{A}_k$) given as

$$\mathbf{H}_k^{(i)} = \left[\begin{array}{c|c|c} \mathbf{H}_{k,11}^{(i)} & \mathbf{0} & \mathbf{H}_{k,12}^{(i)} \\ \hline \mathbf{0} & \mathbf{I}_{(n-1)\nu-r} & \mathbf{0} \\ \hline \mathbf{H}_{k,21}^{(i)} & \mathbf{0} & \mathbf{H}_{k,22}^{(i)} \end{array}\right], \tag{9}$$

where $\mathbf{H}_{k,11}^{(i)}$ is $r \times r$ identity matrix except for the ith diagonal entry, $\mathbf{H}_{k,12}^{(i)}$ is $r \times \nu$ zero matrix except for the ith row, $\mathbf{H}_{k,12}^{(i)} = \mathbf{H}_{k,21}^{(i)}$, and $\mathbf{H}_{k,22}^{(i)}$ is a symmetric $\nu \times \nu$ matrix.

It is thus we have $\mathbf{H}_k\overline{\mathbf{Q}}_{k-1}\dot{\mathbf{A}}_k = \left[\begin{array}{c} \mathbf{R}_k \\ \hline \mathbf{0} \end{array}\right]$ and $\mathbf{Q}_k = \mathbf{H}_k\overline{\mathbf{Q}}_{k-1}$. Now with

$$\mathbf{Q}_k\left[\begin{array}{c|c} \dot{\mathbf{A}}_k & \dot{\mathbf{y}}_k^{(f)} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{R}_k & \mathbf{u}_k \\ \mathbf{0} & \mathbf{v}_k \end{array}\right], \tag{10}$$

Table 1: CP-based *Direct* SBHT-RLS algorithm (CPE2).

Input: $\mathbf{y}_k^{(f)}, \mathbf{y}_{k-1}^{(f)}, \ldots, \mathbf{y}_{k-2}^{(f)}(\nu)$ and $\mathbf{Y}_k$

Known parameters: Γ_i and σ_i

Selecting parameters: λ (with $0 < \lambda \leq 1$)

Initialization: $k = 0$, an initial training process is used to initialize $\hat{\mathbf{h}}_0$ and $\mathbf{R}_0 = \delta\mathbf{I}$ (with $0 < \delta \ll 1$ is small positive scalar).

Algorithm: $k = 1, 2, 3, \ldots$

(1) $\widetilde{\mathcal{H}}_{i,k-1} = (1/\sqrt{m})\sum_{l=0}^{\nu}\hat{h}_{l,k-1}e^{-j(2\pi il)/m}$, $i = 0, \ldots, m-1$

(2) $w_{i,k-1} = (\Gamma_i^{1/2}\mathcal{H}_{i,k-1}^*)/(\Gamma_i\|\mathcal{H}_{i,k-1}\|^2 + \sigma_i^2)$, $i = 0, \ldots, m-1$

(3) $\hat{X}_{i,k} = Y_{i,k}w_{i,k-1}$, $i = 0, \ldots, m-1$

(4) $\overline{x}_{i,k} = (1/\sqrt{m})\sum_{l=0}^{m-1} q(\hat{X}_{l,k})e^{j(2\pi il)/m}$, $i = m-\nu, \ldots, m-1$

(5) $\dot{\mathbf{A}}_k = \begin{bmatrix} \sqrt{\lambda}\mathbf{R}_{k-1} \\ \overline{\mathbf{A}}_k \end{bmatrix}, \dot{\mathbf{y}}_k^{(f)} = \begin{bmatrix} \lambda\mathbf{y}_{k-2}^{(f)}(\nu) \\ \sqrt{\lambda}\mathbf{y}_{k-1}^{(f)} \\ \mathbf{y}_k^{(f)} \end{bmatrix} = \begin{bmatrix} \sqrt{\lambda}\dot{\mathbf{y}}_{k-1}^{(f)} \\ \mathbf{y}_k^{(f)} \end{bmatrix}$

(6) $\mathbf{H}_k\left[\begin{array}{c|c} \dot{\mathbf{A}}_k & \dot{\mathbf{y}}_k^{(f)} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{R}_k & \mathbf{u}_k \\ \mathbf{0} & \mathbf{v}_k \end{array}\right]$,

where $\mathbf{H}_k = \mathbf{H}_k^{(r)}\mathbf{H}_k^{(r-1)}, \ldots, \mathbf{H}_k^{(1)}$,

with $\mathbf{H}_k^{(i)} = \mathbf{I}_{p-i+1} - \beta\mathbf{v}\mathbf{v}^T$, $p = 2\nu + 1$, $\beta = 2/\mathbf{v}^T\mathbf{v}$,

$\mathbf{v} = \mathbf{x} \pm \|\mathbf{x}\|_2\mathbf{e}_1$,

$\mathbf{e}_1 = [1, 0, \ldots, 0]^T$, and $\mathbf{x} = \dot{\mathbf{A}}_k(i : p, i)$.

(7) Solve $\mathbf{u}_k = \mathbf{R}_k\hat{\mathbf{h}}_k$ through back substitution.

where $\mathbf{u}_k = [u_{0,k}, u_{1,k}, \ldots, u_{\nu,k}]^T$ and

$$\mathbf{R}_k = \begin{pmatrix} r_{(0,0),k} & r_{(0,1),k} & \cdots & r_{(0,\nu),k} \\ 0 & r_{(1,1),k} & \cdots & r_{(1,\nu),k} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{(\nu,\nu),k} \end{pmatrix}, \tag{11}$$

the optimal solution is thus obtained by solving the upper triangular system $\mathbf{R}_k\hat{\mathbf{h}}_k = \mathbf{u}_k$ by back substitution operation as follows:

$$\hat{h}_{i,k} = \frac{u_{i,k} - \sum_{j=i+1}^{\nu} r_{(i,j),k}\hat{h}_{j,k}}{r_{(i,i),k}}, \quad i = \nu, \ldots, 0. \tag{12}$$

The matrix $\dot{\mathbf{A}}_{k-1}$ can be uniquely QR factorized only if it is full column rank (i.e., rank $\dot{\mathbf{A}}_{k-1} = r$). Therefore, the minimum number of rows in $\dot{\mathbf{A}}_{k-1}$ must be at least large as the number of columns. To satisfy this requirement and thus to reduce the number of received blocks needed by CPE2 (and CPE3 in Section 3.2), in step (10), we set

$$\dot{\mathbf{A}}_k = \begin{bmatrix} \lambda^{1/2}\mathbf{R}_{k-1} \\ \mathbf{A}_k \end{bmatrix},$$
$$\dot{\mathbf{y}}_k^{(f)} = \Lambda\begin{bmatrix} \mathbf{y}_{k-2}^{(f)}(\nu) = y_{-1,k-2} \\ \mathbf{y}_{k-1}^{(f)} \\ \mathbf{y}_k^{(f)} \end{bmatrix} = \Lambda\begin{bmatrix} \dot{\mathbf{y}}_{k-1}^{(f)} \\ \mathbf{y}_k^{(f)} \end{bmatrix}. \tag{13}$$

Based on the above discussion, CPE2 algorithm is summarized in Table 1.

Table 2: CP-based *Extended* SBHT-RLS algorithm (CPE3).

Input: $\mathbf{y}_k^{(f)}, \mathbf{y}_{k-1}^{(f)}, \ldots, \mathbf{y}_{k-2}^{(f)}(\nu)$ and $\mathbf{Y}_k$
Known parameters: Γ_i and σ_i
Selecting parameters: λ (with $0 < \lambda \leq 1$)
Initialization: $k = 0$, an initial training process is used to initialize $\hat{\mathbf{h}}_0$, $\mathbf{R}_0 = \delta\mathbf{I}$,
$\mathbf{R}_0^{-T} = \delta^{-1}\mathbf{I}$ (with $0 < \delta \ll 1$ is small positive scalar).
Algorithm: $k = 1, 2, 3, \ldots$

(1) $\widetilde{\mathcal{H}}_{i,k-1} = (1/\sqrt{m}) \sum_{l=0}^{\nu} \hat{h}_{l,k-1} e^{-j(2\pi i l)/m}$, $i = 0, \ldots, m-1$
(2) $w_{i,k-1} = (\Gamma_i^{1/2} \mathcal{H}_{i,k-1}^*)/(\Gamma_i \|\mathcal{H}_{i,k-1}\|^2 + \sigma_i^2)$, $i = 0, \ldots, m-1$
(3) $\hat{X}_{i,k} = Y_{i,k} w_{i,k-1}$, $i = 0, \ldots, m-1$
(4) $\overline{x}_{i,k} = (1/\sqrt{m}) \sum_{l=0}^{m-1} q(\hat{X}_{l,k}) e^{j(2\pi i l)/m}$, $i = m-\nu, \ldots, m-1$
(5) $\dot{\mathbf{A}}_k = \begin{bmatrix} \sqrt{\lambda}\mathbf{R}_{k-1} \\ \overline{\mathbf{A}}_k \end{bmatrix}, \dot{\mathbf{B}}_k = \begin{bmatrix} \mathbf{R}_{k-1}^{-T}/\sqrt{\lambda} \\ \mathbf{0}^T \end{bmatrix}$,
$\dot{\mathbf{y}}_k^{(f)} = \begin{bmatrix} \lambda \mathbf{y}_{k-2}^{(f)}(\nu) \\ \sqrt{\lambda}\mathbf{y}_{k-1}^{(f)} \\ \mathbf{y}_k^{(f)} \end{bmatrix} = \begin{bmatrix} \sqrt{\lambda}\dot{\mathbf{y}}_{k-1}^{(f)} \\ \mathbf{y}_k^{(f)} \end{bmatrix}$
(6) $\mathbf{H}_k \left[\dot{\mathbf{A}}_k \,\middle|\, \dot{\mathbf{B}}_k \,\middle|\, \dot{\mathbf{y}}_k^{(f)} \right] = \left[\begin{array}{c|c|c} \mathbf{R}_k & \mathbf{R}_k^{-T} & \mathbf{u}_k \\ \mathbf{0} & \mathbf{W}_k^T & \mathbf{v}_k \end{array} \right]$,
where $\mathbf{H}_k = \mathbf{H}_k^{(r)} \mathbf{H}_k^{(r-1)}, \ldots, \mathbf{H}_k^{(1)}$,
with $\mathbf{H}_k^{(i)} = \mathbf{I}_{p-i+1} - \beta \mathbf{v}\mathbf{v}^T$, $p = 2\nu + 1$, $\beta = 2/\mathbf{v}^T\mathbf{v}$,
$\mathbf{v} = \mathbf{x} \pm \|\mathbf{x}\|_2 \mathbf{e}_1$,
$\mathbf{e}_1 = [1, 0, \ldots, 0]^T$, and $\mathbf{x} = \dot{\mathbf{A}}_k(i : p, i)$.
(7) $\mathbf{h}_k = \mathbf{h}_{k-1} - \mathbf{W}_k \mathbf{v}_k$

3.2. CP-Based Extended SBHT-RLS Algorithm (CPE3). In this section, we propose an alternative approach by appending one more column to the matrices of CPE2 algorithm. To simplify the derivation, we combine the first column of (10) and the new column to construct the formula

$$\mathbf{Q}_k \left[\begin{array}{c|c} \sqrt{\lambda}\mathbf{R}_{k-1} & \mathbf{R}_{k-1}^{-T}/\sqrt{\lambda} \\ \mathbf{A}_k & \mathbf{0}^T \end{array} \right] = \left[\begin{array}{c|c} \mathbf{R}_k & \mathbf{R}_k^{-T} \\ \mathbf{0} & \mathbf{W}_k \end{array} \right]. \tag{14}$$

We next define a lemma, known as the *matrix factorization lemma* [33] that is very elegant tool in the development of QRD-RLS algorithms.

Lemma 1. *If* $\mathbf{A}$ *and* $\mathbf{B}$ *are any two* $N \times M (N \leq M)$ *matrices, then*

$$\mathbf{A}^T\mathbf{A} = \mathbf{B}^T\mathbf{B}, \tag{15}$$

if and only if there exists an $N \times N$ *unitary matrix* $\mathbf{Q}(\mathbf{Q}^T\mathbf{Q} = \mathbf{I})$ *such that*

$$\mathbf{Q}\mathbf{A} = \mathbf{B}. \tag{16}$$

Applying Lemma 1 to (14), we obtain

$$\mathbf{R}_k^{-T}\mathbf{R}_k^T = \mathbf{R}_{k-1}^{-T}\mathbf{R}_{k-1}^T = \mathbf{I}_r. \tag{17}$$

This shows that $\mathbf{R}_k^{-T}$ obtained is the correct inverse transposition of $\mathbf{R}_k^T$ and can be updated by using the same orthonormal transformation $\mathbf{Q}_k$.

Next, we combine the second column of (10) and the new column to construct the formula

$$\mathbf{Q}_k \left[\begin{array}{c|c} \sqrt{\lambda}\dot{\mathbf{y}}_{k-1}^{(f)} & \mathbf{R}_{k-1}^{-T}/\sqrt{\lambda} \\ \mathbf{y}_k^{(f)} & \mathbf{0}^T \end{array} \right] = \left[\begin{array}{c|c} \mathbf{u}_k & \mathbf{R}_k^{-T} \\ \mathbf{v}_k & \mathbf{W}_k^T \end{array} \right]. \tag{18}$$

Now by applying Lemma 1 to (18) yields

$$\mathbf{R}_k^{-1}\mathbf{u}_k + \mathbf{W}_k\mathbf{v}_k = \mathbf{R}_{k-1}^{-1}\mathbf{u}_{k-1}. \tag{19}$$

From (19), we establish a simple recursion to compute the channel vector

$$\mathbf{h}_k = \mathbf{h}_{k-1} - \mathbf{W}_k\mathbf{v}_k. \tag{20}$$

This recursion can be written in component form as

$$h_{i,k} = h_{i,k-1} - \mathbf{w}_{i,k}^T\mathbf{v}_k, \quad i = 0, \ldots, \nu, \tag{21}$$

where $\mathbf{w}_{i,k}$ is the ith column of the matrix $\mathbf{W}_k^T$.

Based on the above discussion, CPE3 is formulated in Table 2.

Remarks. (i) Both algorithms are initialized in a training mode, the algorithms then switch to a decision-directed mode for channel tracking. Note that, in step (1), based on the previous channel estimate $\hat{\mathbf{h}}_{k-1}$, the previous frequency response $\widetilde{\mathcal{H}}_{k-1} = [\mathcal{H}_{0,k-1}, \ldots, \mathcal{H}_{m-1,k-1}]$ is computed. In step (2), $\widetilde{\mathcal{H}}_{k-1}$ is then used to compute equalization coefficients. The decision-directed data vector $\hat{\mathbf{X}}_k$ is then computed in step (3). In step (4), symbol estimates are projected onto the finite alphabet (FA), and the estimated transmitted CP data $\overline{\mathbf{x}}_k^{(f)}$ is obtained by performing partial FFT on the decision-directed projected samples $\overline{\mathbf{X}}_k$. In steps (5) through (7), the new channel estimate is then obtained by treating the resulting symbol estimates as the known symbols. The process of alternating between channel and symbol estimation steps is applied repeatedly.

(ii) In [29], Sakai has derived a method for extracting weight coefficients based on the inverse factorization method of Pan and Plemmons [34] and Liu's SBHT-RLS algorithm. The time updating formula for channel coefficients is obtained by first generalizing the inverse factorizations for the block case and then deriving a formula for updating the channel coefficients. The complicated and challenging derivation gets rid of matrix operations by exploiting the relation between *a priori* and *posteriori* error vectors. Based on [35] and suggested by its author, Sakai has also presented a simpler derivation for updating the channel vector in [30]. In contrast, in the above discussion, the same result is derived by following a straightforward approach by generalizing the *Extended* QRD-RLS algorithm of Yang and Bohme [28] to the block RLS case.

4. Computational Complexity and Systolic Array Implementation

4.1. Computational Complexity. The CPE1, CPE2, and CPE3 algorithms are similar in the CP estimation part (i.e., steps (1) through (4)), we therefore compare their complexities in the channel estimation part. The CPE1 channel estimation stage requires $\mathcal{O}(r^3)$ computations to update $\mathbf{h}$. In contrast, due to absence of any matrix inversion as opposed to CPE1, it is possible to implement channel estimation parts of both the algorithms with $\mathcal{O}(r^2)$ operations per time update. This indicates that the proposed algorithms are computationally superior than the CPE1.

4.2. Systolic Array Implementation. The detection part of both proposed algorithms (comprising of steps (1) through (4)) is particularly simple for which many efficient systolic array architectures have been proposed. We therefore limit our discussion to possible implementation architectures for channel estimation part of the proposed algorithms.

The systolic array implementation of channel estimation section of CPE2 and its processing cells are shown in Figure 2, where adaptive filtering triangular update part (comprising of step (6)) is realized on a triangular vectorial systolic array as in [24] for $\mathbf{R}_k$ and $\mathbf{u}_k$ extraction. It consists of two sections: the upper triangular array (shown in part (a) of Figure 2), which stores and updates $\mathbf{R}_k$ and the right-hand column of cells (shown in part (b) of Figure 2), which stores and updates $\mathbf{u}_k$. The input data are fed from top and propagate to the bottom of the array. The rotation angles are calculated in left boundary cells, and propagate from left to right. The resulting $\mathbf{R}_k$ and $\mathbf{u}_k$ updates in step (6) are subsequently used in the linear bidirectional systolic array section [19] (shown in part (c) of Figure 2) to obtain the channel estimate using back substitution operation. Unfortunately, a critical obstruction appears because the process of the triangular-updates runs from the upper-left corner to the lower-right corner of the array, while the process of the back substitution runs in exactly the opposite direction. It is therefore pipelining of the two steps (the triangular update and back substitution) that seems impossible on a triangular array. Back substitution may be implemented as a separate operation on a parallel two-dimensional array [36]. Nevertheless, the two-dimensional array can become quite large for long channel lengths, requiring a substantial area for VLSI implementation. On the other hand, comparatively simpler linear array structure shown in Figure 2 is highly sequential, thus involving more time delay due to increased clock cycles to compute the channel coefficients. For these reasons, the back substitution in CPE2 is a costly operation to perform.

The CPE3 approach involves a time recursive QR solution to compute the channel vector $\mathbf{h}$. The channel estimation part of CPE3 algorithm can be implemented by a fully pipelined rhombic systolic array obtained by combining lower triangular array with an upper triangular array. This implementation has been performed by Sakai in [29, 30] and is reproduced in Figure 3. The components of $\mathbf{R}_k$ are updated in the upper triangular part (a) of Figure 3. Also, the components of $\mathbf{u}_k$ are updated in part (b) in the same fashion as the off-diagonal components of $\mathbf{R}_k$, with the input data $\mathbf{y}_k$ from the top of this column and the output $\mathbf{v}_k$ from the bottom of this column. Notice that systolic implementation in upper section of part (c) is similar to that in part (a), except that the array is now lower triangular, and each element is divided with $\beta = \sqrt{\lambda}$ before updating, and the input to the array is provided from the top in the form of a zero vector. A systolic array performing (20) is shown in lower portion of part (c) of Figure 3, where the cells in the bottom line, shown by small circles; perform (20) for calculating the tap coefficients. Each column of the lower triangular array whose cells are shown by diamonds perform $\mathbf{R}_k^{-T}$ updating. The cells also calculate each column of $\mathbf{U}_k^T$, appearing from the last diamond cell. Notice that due to absence of back substitution, the CPE3 algorithm is rich in parallel operations and therefore leads to more efficient and simple implementation on systolic processors.

5. Simulation Results

In this section, floating-point and fixed-point simulation results are presented to examine and compare the performance of the CPE1, CPE2, and CPE3 approaches. All simulations were carried out in a typical asymmetric digital subscriber line (ADSL) environment with perfect block synchronization, FFT size $m = 512$, the CP length $\nu = 32$, $\lambda = 0.75$, $\delta = 1e^{-3}$, and 4-QAM constellation for modulation, unless otherwise stated. For a fair comparison, for CPE1 we set forgetting factor across blocks $\mu_1 = \lambda$ and forgetting factor within blocks $\mu_2 = 1$. The mismatch performance is evaluated by averaged mean-square-error (MSE) per subchannel err $= \sum_{i\in U} \|X_i - \hat{X}_i\|/|U|$, where U is the set of indexes corresponding to the U used subchannels and $|U|$ is the number of all the used subchannels [10]. The transmit power of all used subchannels is same (i.e., $\sigma_i^2 = \sigma^2$) and the noise power was set such that SNR= 30 dB (a typical value of SNR in ADSL environments).

The discrete channel impulse response with transfer function $H0(D)$ for carrier service loop area (CSA) loop # 1 was obtained from the Matlab DMTTEQ Toolbox [37] and sampled at 2.208 MHz. For simulation purposes, the shorter channel was generated by subsampling. $H0(D)$ was perturbed to obtain another test channel $H1(D)$ (to mimic small variation in $H0(D)$). Corresponding frequency responses for the two test channels are shown in Figure 4. Initially, the channel transfer function is $H0(D)$, which remains unchanged for the first 400 data blocks. At data block 401, the channel is switched from $H0(D)$ to $H1(D)$. For all adaptive schemes, only the first DMT symbol was sent as pure training sequence to identify the initial channel for fast convergence. Also, the inverse of the correlation matrix in CPE1 is initialized to a constant multiple of the identity matrix.

Example 1. Figure 5 shows typical learning curves of the three algorithms, with adaptation factor parameter λ values

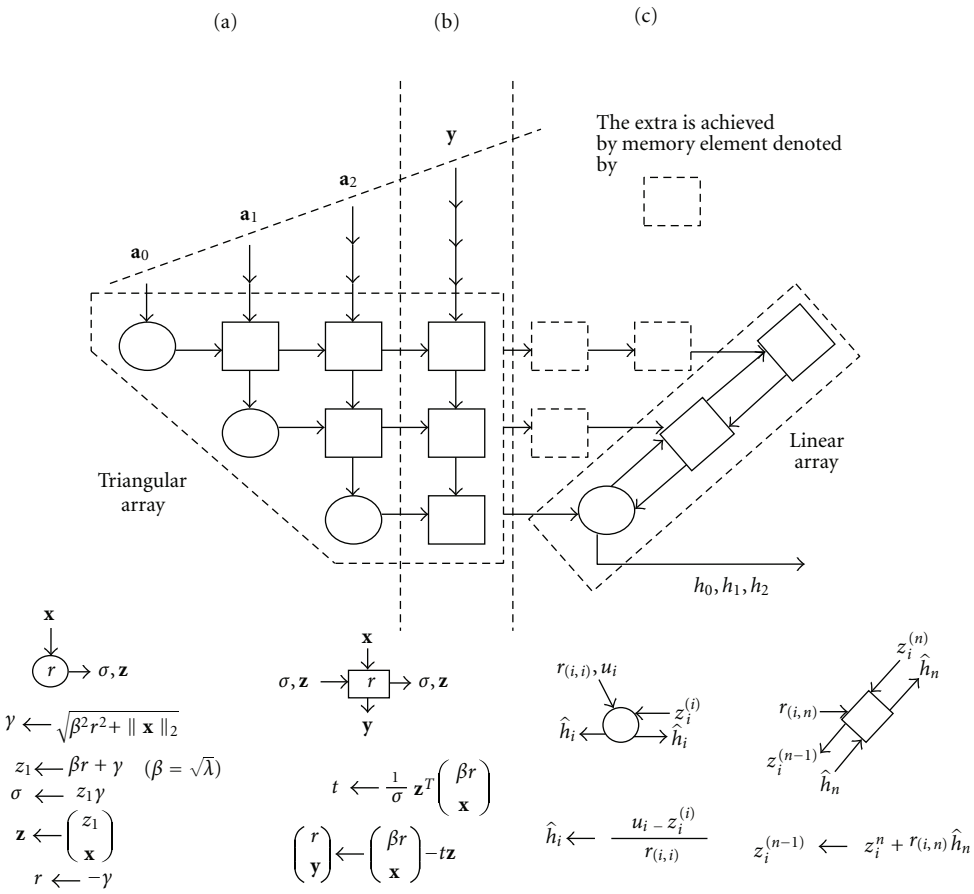

Figure 2: Systolic array implementation of channel estimation section of CPE2 (using Householder transformations) with processing cell descriptions.

of 0.75 (top plots) and 0.55 (bottom plots), under double-precision floating-point implementation (using IEEE standard for floating-point arithmetic (IEEE 754)). It can be seen that all the schemes are able to converge and can track the channel variation. The learning curves of CPE1 and CPE2 are overlaid and both the algorithms converge faster than CPE3. As compared to CPE3, the two algorithms are also seen to have greater uneven performance. In contrast, although CPE3 convergence is slower, it is seen to demonstrate superior steady-state performance. A close examination of CEP2 algorithm shows that the back substitution operation involves decision-feedback computation of channel coefficients. If a channel coefficient suffers from an error, this error weights heavily in the estimation of the next and subsequent channel coefficients. The erroneous estimated channel causes the next detection error. This decision error further propagates and causes subsequent decision errors. Consequently, CPE2 encounters performance loss. In contrast, channel is recursively updated without back substitution in CPE3. CPE3 is therefore seen to yield better performance.

A close observation of top and bottom plots of Figure 5 also indicates that convergence rate and steady-state performance of the three algorithms can be improved by lowering the value of λ. The price paid in growth is uneven performance which can be reduced and thus numerical stability can be improved by increasing the data block size (i.e., with the increased CP length), while the system latency is increased.

Example 2. Without giving a rigorous stability analysis, we verify the stability of the CPE1, CPE2, and CPE3 algorithms experimentally through a long-time simulation with 5×10^3 data blocks (considerably large number of samples). Corresponding results in Figure 6 show that the three algorithms do not show any sign of divergence and have very stable performance.

Example 3. The more complex the modulation alphabet, the narrower the gap between the symbol decision space and the higher the probability of error in detecting the signal [38]. Since the three algorithms rely on the FA property of source symbols, high-performance degradation is expected as the constellation size increases. It is therefore the three algorithms that may not be suitable for rate adaptation. To verify this, in this simulation example, we repeat Example 1 with 16-QAM and 64-QAM constellation sizes. Corresponding simulation results in Figure 7 show that the three algorithms

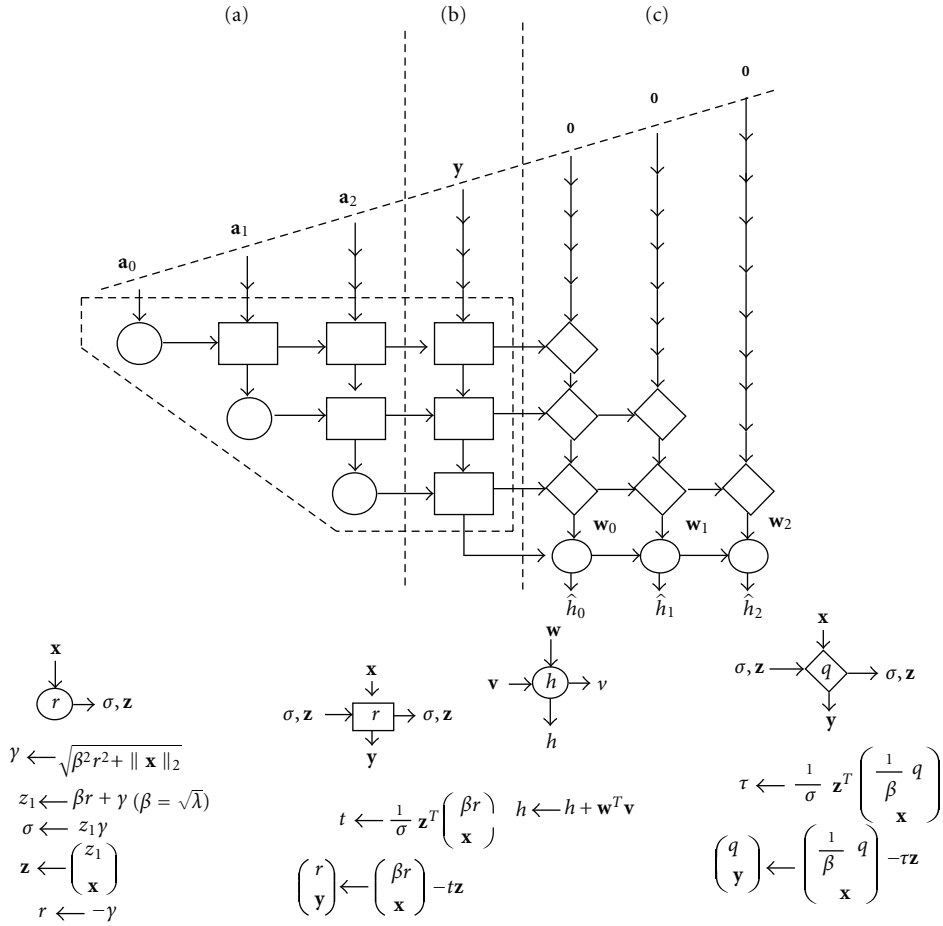

Figure 3: Systolic array implementation of channel estimation section of CPE3 (using Householder transformations) with processing cell descriptions.

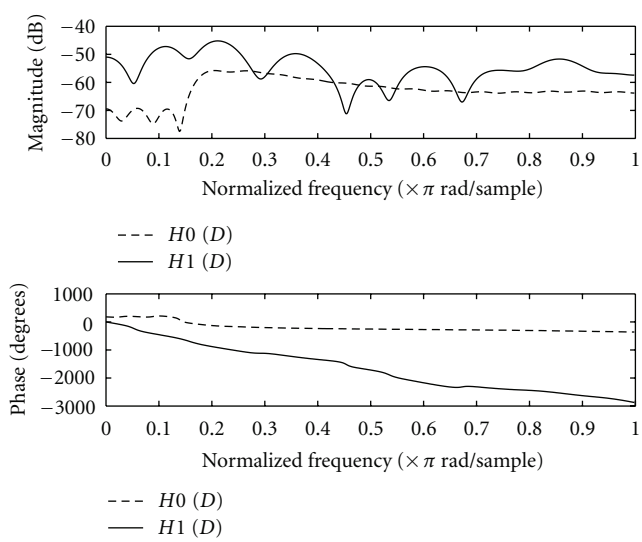

Figure 4: Frequency responses for channels $H0(D)$ and $H1(D)$.

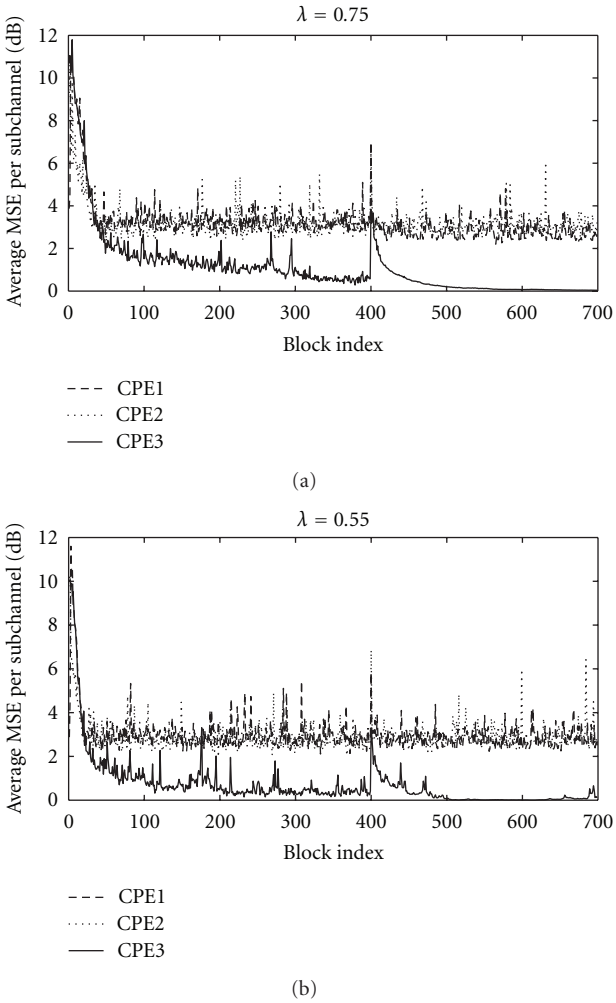

Figure 5: Effect of the forgetting factor λ on performance analysis curves of CPE1, CPE2, and CPE3 algorithms when tracking channels $H0(D)$ and $H1(D)$: $\lambda = 0.75$ (top plots) and $\lambda = 0.55$ (bottom plots).

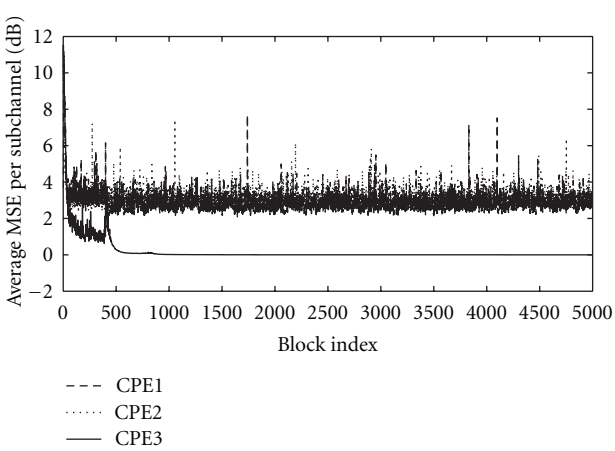

Figure 6: Stability performance comparison with 5×10^3 data blocks ($\lambda = 0.75$ and 4-QAM).

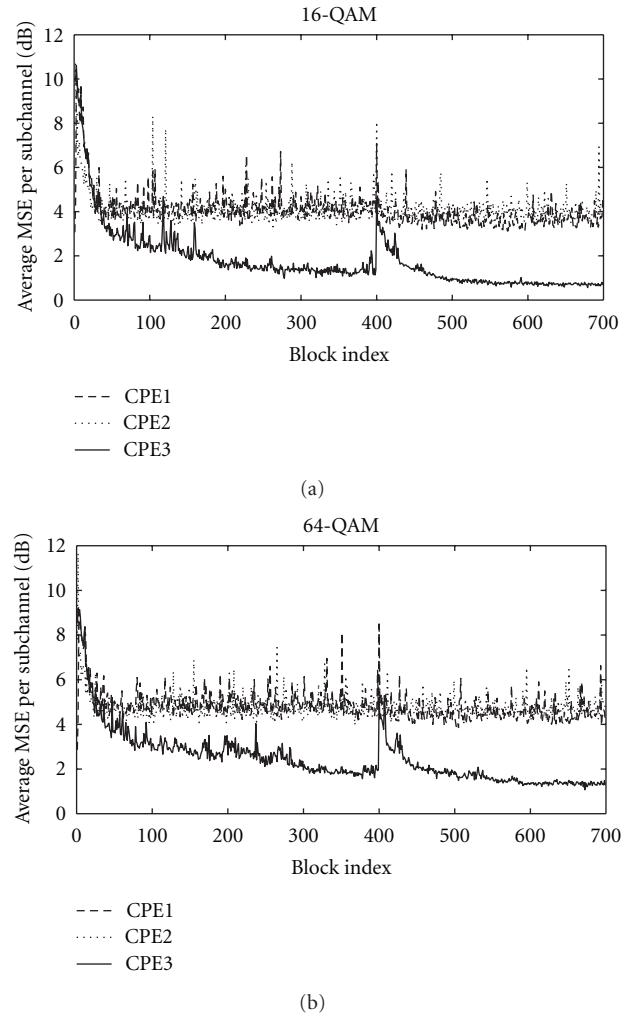

Figure 7: Effect of modulation constellation size on algorithm performance curves (16-QAM (top plots), 64-QAM (bottom plots), $\lambda = 0.75$).

take the same number of data blocks to converge. However, as expected, their performance degrades when the constellation size is increased.

Example 4. In this section, due to inherent parallelism and thus suitability for fixed-point VLSI implementation, we examine the fixed-point performance of the CPE2 and CPE3 algorithms with 16, 24, and 32 bit data word length W_L implementations for both data and channel coefficients. These $W_L s$ are selected as a reasonable approximation as these data lengths are suitable for many applications. For fixed-point simulations, routines in Matlab are developed to mimic the operations of fixed-point arithmetic, and all quantities in the algorithms are represented with finite bits. The fixed-point representation requires $W_L = (l_i$ bits for integer part) + (l bits for the fractional part) + (1 bit for sign). For real number x, its quantized value x^q is obtained as follows. With W_L bits, the largest integers that can be represented are $\pm 2^{(W_L-1)} - 1$. When the value of x falls outside

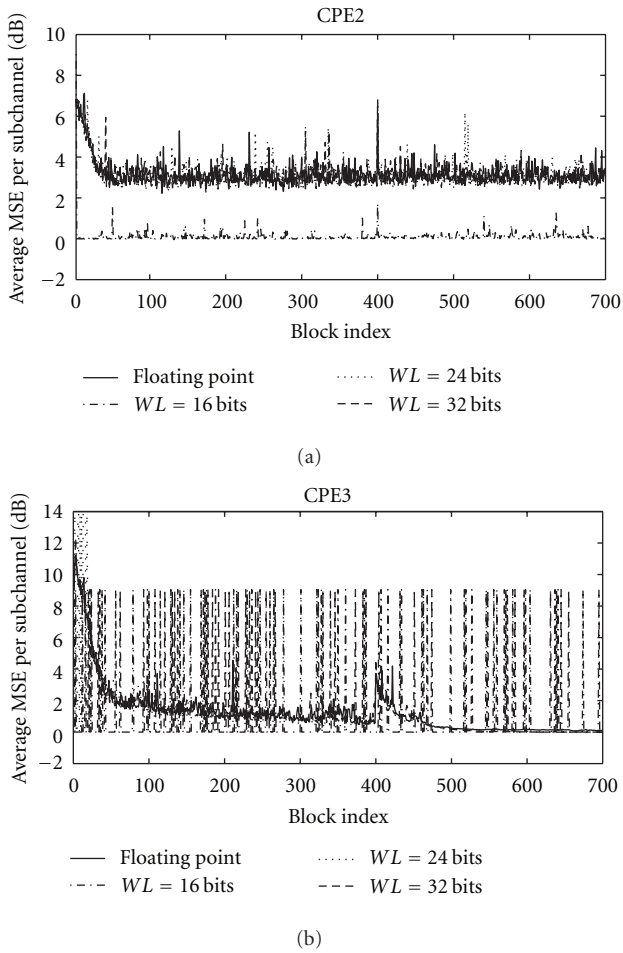

Figure 8: Plots of fixed-point performance of CPE2 (top) and CPE3 (bottom) with three choices of quantization bits ($\lambda = 0.75$, 4-QAM).

the interval $[+2^{(W_L-1)}-1, -2^{(W_L-1)-1}]$, the saturation occurs, and the x^q is then taken as one of the boundary values, $+2^{(W_L-1)}-1$ or $-2^{(W_L-1)}-1$. On the other hand, if x lies within the interval $[+2^{(W_L-1)}-1, -2^{(W_L-1)}-1]$, then the l_i bits are computed to represent the integer part of x, and the remaining bits l are used to represent the fractional part of x. It is important to note here that for the above choice of $W_L s$, the thresholds are sufficiently larger than signal values involved in both the algorithms. The quantizer is therefore always expected to operate on values that are much lesser than the boundary values, and therefore no saturation errors are expected. The only errors that are introduced by finite precision approximations are the round-off errors.

Figure 8 provides performance plots of CPE2 (top) and CPE3 (bottom) with different W_L choices and floating-point performance. From the performance curves, we infer that both algorithms are able to track the channel without numerical stability issues with $W_L s$ 24 and 32. The performance curves with W_L of 16 bits indicate unacceptable performance or breakdown caused by quantization errors for both the algorithms. For both the algorithms, increasing W_L above 24 bits does not result in any improvement and performance curves of their 24-bit and floating-point implementations are overlaid (there is no visible difference). It is therefore 24-bit finite word implementation is a reasonable approximation of their floating-point computation.

6. Conclusion

In this paper, by using numerically robust block HTs, two CP-based adaptive channel estimation algorithms have been presented for multicarrier systems. Conceptually, the new schemes maintain the same spirit of the CP based CMB-RLS channel tracking scheme. More precisely, the basic idea is to utilize CP data from the data detection part for adaptive channel estimation. The new approaches achieve the same purpose by replacing the computationally expensive CMB-RLS channel estimation part with the computationally cheaper SBHT-RLS alternatives. Among the two schemes, the method called CP based *Direct* SBHT-RLS is based upon Liu's algorithm in the channel estimation part but adaptively updates channel vector instead of the error vector. The second method called CP based *Extended* SBHT-RLS is based upon Sakai's algorithm in the channel estimation part but uses an independent and simpler derivation.

Floating-point performance curves indicate that all the three schemes are able to converge and can track channel variation without any stability problems. CPE1 and CPE2 exhibit identical stable performance, whereas CPE3 outperforms both the CPE1 and CPE2 techniques. In contrast to CPE1, what is remarkable here is that the CPE2 and CPE3 algorithms achieve their performance at lower computational complexity, enhanced parallelism, and pipelining for systolic array/VLSI implementation. All the three algorithms are seen to converge faster and perform better with lower values of forgetting factor parameter λ. Our simulation results suggest that such advantages come at the price of greater uneven performance. Hence, moderate values of forgetting factor would be preferred where a balance in both performance and stablility is required. The three techniques also show reduction in performance with the increase in modulation constellation size. Hence, these techniques are more appealing when the constellation size is small and may not be suitable for rate adaptation. It is also shown that in terms of finite word length behavior, 24-bit finite word implementation is a reasonable approximation of their typical floating-point computation (In practice, the word lengths are optimized with respect to the actual system requirements (i.e., chip area, latency, power consumption, FFT size, throughput), noise, channel length, and desired acceptable performance.).

Systolic array structures that allow efficient parallel implementations of the schemes with VLSI technology in real time were considered. The CPE2 approach is partially concurrent due to costly back substitution operation, whereas, CPE3 approach is highly concurrent due to the absence of back substitution operation and therefore lead to more efficient implementation on systolic processors.

The methods proposed in this paper are well suited for applications where good numerical properties, computational saving, and parallel processing implementation

advantages (with improved performance (in case of CPE3 only)) are desired. Although a real baseband DMT case is the main focus of this paper, the proposed approaches can also be applied to the complex baseband case (wireless multicarrier systems). In such case, a further improvement in performance is possible by including forward error correction (FEC) decoding in the reliable reconstruction of transmitted symbols. Future interesting directions include studying hardware implementation problems, fine grain implementation/architecture of processing elements to workout total cost of operators (adders, multipliers, dividers, memory elements (delay elements), etc.) and algorithm latencies, modifications of the schemes to achieve reduced complexity, performance improvement, and stable implementations with reduced word lengths.

Acknowledgment

The author wishes to express his sincere thanks to the reviewers for their constructive comments and useful suggestions towards improving this paper.

References

[1] Z. Wang and G. B. Giannakis, "Wireless multicarrier communications: where Fourier meets Shannon," *IEEE Signal Processing Magazine*, vol. 17, no. 3, pp. 29–48, 2000.

[2] "IEEE Part II: wireless LAN medium access control (MAC) and physical layer (PHY) specifications: high speed physical layer in 5 GHz band," IEEE Std. 802.11a-1999, September 1999.

[3] K. Van Acker, G. Leus, M. Moonen, O. van de Wiel, and T. Pollet,"Per tone equalization for DMT-based systems," *IEEE Transactions on Communications*, vol. 49, no. 1, pp. 109–119, 2001.

[4] J. S. Chow, J. C. Tu, and J. M. Cioffi, "A discrete multitone transceiver system for HDSL applications," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 6, pp. 895–908, 1991.

[5] "Radio broadcasting systems: digital audio broadcasting (DAB) to mobile, portable and fixed receivers," ETSI ETS 300 401, 1.3.2 ed., 2000.

[6] "Digital video broadcasting (DVB): framing structure, channel coding and modulation for digital terrestrial television," ETSI EN 300 744, 1.3.1 ed., 2000.

[7] R. van Nee, G. Awater, M. Morikura, H. Takanashi, M. Webster, and K. W. Halford, "New high-rate wireless LAN standards," *IEEE Communications Magazine*, vol. 37, no. 12, pp. 82–88, 1999.

[8] P. J. W. Melsa, R. C. Younce, and C. E. Rohrs, "Impulse response shortening for discrete multitone transceivers," *IEEE Transactions on Communications*, vol. 44, no. 12, pp. 1662–1672, 1996.

[9] P. S. Chow, J. M. Cioffi, and J. A. C. Bingham, "A Practical discrete multitone transceiver loading algorithm for data transmission over spectrally shaped channels," *IEEE Transactions on Communications*, vol. 43, no. 234, pp. 773–775, 1995.

[10] X. Wang and K. J. R. Liu, "Adaptive channel estimation using cyclic prefix in multicarrier modulation system," *IEEE Communications Letters*, vol. 3, no. 10, pp. 291–293, 1999.

[11] X. Wang and K. J. R. Liu, "Performance analysis for adaptive channel estimation exploiting cyclic prefix in multicarrier modulation systems," *IEEE Transactions on Communications*, vol. 51, no. 1, pp. 94–105, 2003.

[12] H. Ali and E. P. Ling, "On the performance of CP based exponentially weighted block RLS channel estimation algorithm for OFDM systems," in *Proceedings of IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing (PacRim '09)*, pp. 135–139, Victoria, Canada, August 2009.

[13] W. A. Syafe, K. Nishijo, Y. Nagao, M. Kurosaki, and H. Ochi, "Adaptive channel estimation using cyclic prefix for single carrier wireless system with FDE," in *Proceedings of the 10th International Conference on Advanced Communication Technology*, pp. 1032–1035, February 2008.

[14] D. G. Manolakis, V. K. Ingle, and S. M. Kogon, *Statistical and Adaptive Signal Processing: Spectral Estimation, Signal Modeling, Adaptive Filtering and Array Processing*, McGraw-Hill Education, Singapore, 2000.

[15] K. J. Raghunath and K. K. Parhi, "Finite-precision error analysis of QRD-RLS and STAR-RLS adaptive filters," *IEEE Transactions on Signal Processing*, vol. 45, no. 5, pp. 1193–1209, 1997.

[16] C. F. T. Tang, K. J. R. Liu, S. F. Hsieh, and K. Yao, "VLSI algorithms and architectures for complex Householder transformation with applications to array processing," *The Journal of VLSI Signal Processing*, vol. 4, no. 1, pp. 53–68, 1992.

[17] H. Leung and S. Haykin, "Stability of recursive QRD-LS algorithms using finite-precision systolic array implementation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 5, pp. 760–763, 1989.

[18] M. G. Siqueira and P. S. R. Diniz, "Infinite precision analysis of the QR-recursive least squares algorithm," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '93)*, vol. 1, pp. 878–881, May 1993.

[19] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Upper Saddle River, NJ, USA, 1996.

[20] G. Lightbody, R. Woods, and R. Walke, "Design of a parameterizable silicon intellectual property core for QR-based RLS filtering," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 4, pp. 659–678, 2003.

[21] J. G. McWhirter and T. J. Shepherd, "Systolic array processor for MVDR beamforming," *IEE Proceedings. Part F*, vol. 136, no. 2, pp. 75–80, 1989.

[22] C. R. Ward, P. J. Hargrave, and J. G. McWhirter, "A novel algorithm and architecture for adaptive digital beamforming," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 338–346, 1986.

[23] S. Z. Kalson and K. Yao, "A class of least-squares filtering and identification algorithms with systolic array architectures," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 43–52, 1991.

[24] K. J. R. Liu, S. F. Hsieh, and K. Yao, "Systolic block Householder transformation for RLS algorithm with two-level pipelined implementation," *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 946–958, 1992.

[25] K. J. R. Liu, S. F. Hsieh, and K. Yao, "Recursive LS filtering using block Householder transformations," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP '90)*, vol. 3, pp. 1631–1634, Albuquerque, NM, USA, April 1990.

[26] A. Elnashar, S. Elnoubi, and H. A. El-Mikati, "Performance analysis of blind adaptive MOE multiuser receivers using inverse QRD-RLS algorithm," *IEEE Transactions on Circuits and Systems I*, vol. 55, no. 1, pp. 398–411, 2008.

[27] S.-J. Chern and C.-Y. Chang, "Adaptive linearly constrained inverse QRD-RLS beamforming algorithm for moving jammers suppression," *IEEE Transactions on Antennas and Propagation*, vol. 50, no. 8, pp. 1138–1150, 2002.

[28] B. Yang and J. F. Bohme, "Rotation-based RLS algorithms: unified derivations, numerical properties, and parallel implementations," *IEEE Transactions on Signal Processing*, vol. 40, no. 5, pp. 1151–1167, 1992.

[29] H. Sakai, "A vectorized systolic array for block RLS using inverse factorizations," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '92)*, vol. 4, pp. 233–236, March 1992.

[30] H. Sakai, "A Vectorized systolic array for parallel weight extraction of block RLS," *International Journal of Adaptive Control and Signal Processing*, vol. 8, no. 5, pp. 475–482, 1994.

[31] H. Ali, "A cyclic prefix based adaptive channel estimation algorithm for multicarrier systems," in *Proceedings of the IEEE International Symposium on Signal Processing and Information Technology (ISSPIT '10)*, Luxor, Egypt, December 2010.

[32] H. Ali, "A cyclic prefix based extended QRD-RLS algorithm using block Householder transformation for adaptive channel estimation in multicarrier systems," in *Proceedings of the 3rd International Conference on Signal Acquisition and Processing (ICSAP '11)*, Singapore, February 2011.

[33] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, 3rd edition, 1996.

[34] C. T. Pan and R. J. Plemmons, "Least squares modifications with inverse factorizations: parallel implications," *Journal of Computational and Applied Mathematics*, vol. 27, no. 1-2, pp. 109–127, 1989.

[35] J. G. McWhirter, "Algorithmic engineering in adaptive signal processing," *IEE Proceedings. Part F*, vol. 139, no. 3, pp. 226–232, 1992.

[36] S. Y. Kung, *VLSI Array Processor*, Prentice Hall, Englewood Cliffs, NJ, USA, 1988.

[37] G. Arslan, M. Ding, B. Lu, Z. Shen, and B. L. Evans, "DMTTEQ Toolbox," The University of Texas at Austin, http://users.ece.utexas.edu/~bevans/projects/adsl/dmtteq/dmtteq.html.

[38] H. Ali, A. Doucet, and D. I. Amshah, "GSR: a new genetic algorithm for improving source and channel estimates," *IEEE Transactions on Circuits and Systems I*, vol. 54, no. 5, pp. 1088–1098, 2007.