An inverse QRD-RLS algorithm for linearly constrained minimum variance adaptive filtering

César A. Medina S.*, Raimundo Sampaio-Neto

Center of Telecommunications Studies (CETUC), Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil

ARTICLE INFO

Article history:
Received 30 May 2012
Received in revised form 24 September 2012
Accepted 2 November 2012
Available online 9 November 2012

Keywords:
Linearly constrained minimum variance
IQRD-RLS filtering
Constrained RLS
CDMA
Blind receiver estimation

1. Introduction

QR decomposition based algorithms have gained an important role in wireless communication systems because their numerical stability in limited precision environments [1], reduced dynamic range, plentiful data and task-level parallelism [2] and efficient implementation in systolic arrays [3] or DSP vector architectures [4]. Such characteristics have previously been exploited to develop algorithms for a range of applications such as adaptive receivers, adaptive equalizers and adaptive beamforming, among others [5–12].

Constrained QR decomposition based RLS algorithms have been proposed in [13–15], nevertheless, they are only solutions to the minimum variance distortion-less response (MVDR) filter, i.e., only one constraint is assumed.

In [7], Chern et al. present a constrained inverse QRD-RLS (CIQRD-RLS) solution for the linearly constrained minimum variance (LCMV) filter formed by two stages. The first stage uses the unconstrained IQRD-RLS algorithm [16] to compute the inverse of the autocorrelation matrix of the input signal, while the second stage applies the matrix-inversion-lemma to the results of the first stage to compute the LCMV filter output.

In this paper we propose a CIQRD-RLS algorithm to compute the coefficients of the LCMV filter that, different from previously proposed algorithms, avoids the use of matrix-inversion-lemma to derive the solution and thus, not only one but two unconstrained IQR-RLS decomposition stages are used to update the LCMV filter coefficients. The proposed algorithm results in better numerical accuracy when compared with previously proposed algorithms, as indicated by computer simulations.

This paper is structured as follows. Section 2 describes the linearly constrained minimum variance least-squares filtering problem, while in Section 3 the proposed algorithm is derived. Some simulation experiments are presented in Section 4, while Section 5 gives the conclusions.

Notation: In what follows, $I_k$ represents a $k \times k$ identity matrix, $0_{m \times n}$, an $m \times n$ null matrix, $\text{diag}(\mathbf{x})$ is a diagonal
matrix with the components of \( \mathbf{x} \) as its nonzero elements, \((\cdot)^T\), \((\cdot)^H\) and \((\cdot)^*\) denote transpose, Hermitian transpose and complex conjugated, respectively.

2. The linearly constrained minimum variance least-squares filter

A linearly constrained minimum variance least-squares filter is shown in Fig. 1, where it is supposed that the samples of the input vector, \( \mathbf{r}(i) \), are drawn from the complex field and, in order to deal with multidimensional input signals, they do not necessarily present a time shift relationship.

The set of filter coefficients \( \mathbf{w}(i) = [w_0(i), w_1(i), \ldots, w_{M-1}(i)]^T \) is chosen in order to minimize the output error in the weighted least-squares sense, subject to a set of \( L \) linear constraints, i.e.,

\[
\mathbf{w}(i) = \arg \min_{\mathbf{w}} \sum_{j=1}^{i} \lambda^{i-j} |\mathbf{w}^H \mathbf{r}(j)|^2 \quad \text{subject to} \quad \mathbf{C}^H \mathbf{w}(i) = \mathbf{p} \tag{1}
\]

where \( 0 < \lambda < 1 \) is the so-called forgetting factor, \( \mathbf{C} \) is the \( M \times L \) constraint matrix and \( \mathbf{p} \) is the \( L \)-element response vector.

Using Lagrange multipliers, the optimum filter coefficients, are obtained as [17]

\[
\mathbf{w}(i) = \mathbf{R}^{-1}(i) \mathbf{C} (\mathbf{C}^H \mathbf{R}^{-1}(i) \mathbf{C})^{-1} \mathbf{p} \tag{2}
\]

where

\[
\mathbf{R}(i) = \sum_{j=1}^{i} \lambda^{i-j} \mathbf{r}(j) \mathbf{r}^H(j) = \lambda \mathbf{R}(i-1) + \mathbf{r}(i) \mathbf{r}^H(i) \tag{3}
\]

is the so-called deterministic correlation matrix of the input signal [3]. Note that (3) is a rank-one update process characterized by the plus sign in front of the rank-one matrix \( \mathbf{r}(i) \mathbf{r}^H(i) \).

3. A constrained inverse QRD-RLS algorithm

The direct implementation of (2) is computationally intensive because involves, among other matrix operations, two matrix inversions. An implementation based on the matrix-inversion-lemma is proposed in [17], leading to the constrained fast recursive least-squares (LCMV FRLS) algorithm. Nevertheless, it is known that such approach leads to a numerically unstable algorithm for finite precision environments. In [7] a CIQR-RLS algorithm was developed in two stages, the first stage uses the (unconstrained) IQRD-RLS algorithm to compute recursively the Cholesky factors of \( \mathbf{R}^{-1}(i) \), and then, in the second stage, the results of the matrix-inversion-lemma are used to compute recursively \((\mathbf{C}^H \mathbf{R}^{-1}(i) \mathbf{C})^{-1}\).

In this paper we propose a CIQR-RLS algorithm implemented in two stages. The first stage updates the Cholesky factors of \( \mathbf{R}^{-1}(i) \) as in [7]. However, differently from [7], the second stage updates \((\mathbf{C}^H \mathbf{R}^{-1}(i) \mathbf{C})^{-1}\) by downdating its Cholesky factors using QR decomposition based algorithms, and thus, leading to a new constrained IQRD-RLS (CIQRD-RLS) algorithm. The proposed algorithm is derived in the following.

3.1. The first stage: updating the \( M \times M \) matrix \( \mathbf{R}^{-1}(i) \)

The (unconstrained) IQRD-RLS algorithm [16] was originally proposed to compute the weight vector of the unconstrained RLS adaptive filter, avoiding recursive matrix inversion or back-substitution as in the conventional RLS algorithm or the QRD-RLS algorithm, respectively. In the first stage of the algorithm proposed here, we update the Cholesky factors of \( \mathbf{R}^{-1}(i) \) using the unconstrained IQRD-RLS algorithm, which we summarize here for completeness of the paper. Note that this stage is also implemented by the CIQRD-RLS algorithm of [7].

Let \( \mathbf{R}(i) \) be a rank-one updated process as in (3) and \( \mathbf{U}(i) \) be its the Cholesky factor, such that \( \mathbf{R}(i) = \mathbf{U}(i)^H \mathbf{U}(i) \), then, it is well known that \( \mathbf{U}(i) \) can be updated from \( \mathbf{U}(i-1) \) as [16]

\[
\begin{bmatrix}
\mathbf{0}_{M \times 1} \\
\mathbf{U}(i)
\end{bmatrix} = \mathbf{Q}(i) \begin{bmatrix}
\mathbf{r}^H(i) \\
\lambda^{1/2} \mathbf{U}(i-1)
\end{bmatrix} \tag{4}
\]

where \( \mathbf{Q}(i) = \mathbf{Q}_{M-1}(i) \mathbf{Q}_{M-2}(i) \cdots \mathbf{Q}_0(i) \) is an orthogonal matrix constructed by a sequence of Givens rotations matrices that annihilates the elements of \( \mathbf{r}^H(i) \) over the diagonal elements of \( \lambda^{1/2} \mathbf{U}(i-1) \). Let express \( \mathbf{Q}(i) \) as

\[
\mathbf{Q}(i) = \begin{bmatrix}
\gamma(i) & \mathbf{g}^H(i) \\
\mathbf{f}(i) & \mathbf{E}(i)
\end{bmatrix} \tag{5}
\]

then, by inverting and conjugate-transposing both sides of the square matrix

\[
\begin{bmatrix}
\mathbf{0}_{M \times 1}^T \\
\mathbf{U}(i)^T & \gamma(i)
\end{bmatrix} = \mathbf{Q}^H(i) \begin{bmatrix}
\mathbf{r}^H(i) & 1 \\
\lambda^{1/2} \mathbf{U}(i-1) & \mathbf{0}_{M \times 1}
\end{bmatrix} \tag{6}
\]
we get

\[
\begin{bmatrix}
    z^H(i) \\
    U^{-H}(i)
\end{bmatrix} = \begin{bmatrix}
    1 / \gamma(i) \\
    0_{M \times 1}
\end{bmatrix} = Q(i) \begin{bmatrix}
    1 \\
    -\gamma(i)
\end{bmatrix} \lambda^{-1/2} U^{-H}(i-1) - a(i)
\]

(7)

where

\[
z(i) = -U^{-1}(i) \gamma(i) \gamma(i)
\]

and

\[
a(i) = \lambda^{-1/2} U^{-H}(i-1) r(i)
\]

Then, to update \( U^{-1}(i) \) from \( U^{-1}(i-1) \) we first compute \( a(i) \) as before, then compute the sequence of Givens rotations matrices, \( Q(i) = Q_{M-1}(i)Q_{M-2}(i) \cdots Q_0(i) \), such that

\[
\begin{bmatrix}
    1 / \gamma(i) \\
    0_{M \times 1}
\end{bmatrix} = Q(i) \begin{bmatrix}
    1 \\
    -\gamma(i)
\end{bmatrix}
\]

(9)

and finally, we apply to

\[
\begin{bmatrix}
    z^H(i) \\
    U^{-H}(i)
\end{bmatrix} = \begin{bmatrix}
    0_{M \times 1} \\
    \lambda^{-1/2} U^{-H}(i-1)
\end{bmatrix}
\]

(10)

such that \( U^{-H}(i) \) is updated from \( U^{-H}(i-1) \).

3.2. The second stage: downdating the \( L \times L \) matrix \( C^{-1}R^{-1}(i)C^{-1} \)

First note that as matrix \( Q(i) \) is orthogonal and since \( R^{-1}(i) = U^{-1}(i)U^{-H}(i) \), we have from (10) that

\[
U^{-1}(i)U^{-H}(i) + z(i)z^H(i) = \lambda^{-1} U^{-1}(i-1)U^{-H}(i-1)
\]

R

\[
R^{-1}(i) = \lambda^{-1} R^{-1}(i-1) - z(i)z^H(i)
\]

(11)

now, pre-multiplying (11) by \( C^H \) and post-multiplying by \( C \) we get that \( \Omega(i) = C^H R^{-1}(i) C \) can be recursively computed as

\[
\Omega(i) = \lambda^{-1} \Omega(i-1) - z(i)z^H(i)
\]

(12)

where

\[
z(i) = C^H z(i)
\]

The key strategy in the proposed algorithm, which allows us to devise a QR decomposition based solution, is to note that (12), different from (3), implements a rank-one downdate process, characterized by the minus sign in front of \( z(i)z^H(i) \), and thus, well known downdating algorithms based on QR decomposition can be used.

There are in the literature several approaches for rank-one downdating process using QR decomposition algorithms, such as the ones proposed in [18–20] that use Givens rotations, [4,21] that use hyperbolic plane rotations and [22] that uses hyperbolic Householder transforms. In all the aforementioned algorithms the forgetting factor is equal to one, \( \lambda = 1 \).

In this paper we use an algorithm based on [4,16] to downdate the inverse of the Cholesky factor of \( \Omega(i) \), for \( \lambda \neq 1 \). Although its derivation is rather straightforward, the authors did not find this more general version in the literature.

In the following, we say that a matrix \( P(i) \) is pseudo-orthogonal if \( P^H(i) \Phi P(i) = \Phi \) for some signature matrix \( \Phi = \text{diag}(\pm 1) \) [23].

Now, let us define \( D(i) \) as the Cholesky factor of \( \Omega(i) \) such that \( \Omega(i) = D^H(i)D(i) \). The key observation here is that a sequence of hyperbolic Householder transforms can be found, so that [22]

\[
\begin{bmatrix}
    D(i) \\
    0_{L \times 1}^T
\end{bmatrix} = \begin{bmatrix}
    P(i) \\
    0_{L \times 1}^T
\end{bmatrix} \begin{bmatrix}
    \lambda^{-1/2} D(i-1) \\
    0_{L \times 1}^T
\end{bmatrix}
\]

(14)

where \( P(i) = P_{L-1}(i) P_{L-2}(i) \cdots P_0(i) \) is a pseudo-orthogonal matrix with respect to \( \Phi = \text{diag}(1,-1) \), such that \( P^H(i) \Phi P(i) = \Phi \) and the downdated Cholesky factor \( D(i) \) satisfies (12). The matrices \( P_s(i) (j = L-1,L-2,...,0) \) are also pseudo-orthogonal and each one annihilates one element of \( z(i) \) over the diagonal elements of \( \lambda^{-1/2} D(i-1) \).

Now, lets partition matrix \( P(i) \) as

\[
P(i) = \begin{bmatrix}
    A(i) & v(i) \\
    t^H(i) & q(i)
\end{bmatrix}
\]

(15)

then, by inverting and conjugate-transposing both sides of the square matrix:

\[
\begin{bmatrix}
    D(i) \\
    0_{L \times 1}^T
\end{bmatrix} = \begin{bmatrix}
    P(i) \\
    0_{L \times 1}^T
\end{bmatrix} \begin{bmatrix}
    \lambda^{-1/2} D(i-1) \\
    0_{L \times 1}^T
\end{bmatrix}
\]

(16)

we get

\[
\begin{bmatrix}
    D^H(i) \\
    0_{L \times 1}^T
\end{bmatrix} = \begin{bmatrix}
    P(i) \\
    0_{L \times 1}^T
\end{bmatrix} \begin{bmatrix}
    \lambda^{1/2} D^{-H}(i-1) - \beta(i) \\
    0_{L \times 1}^T
\end{bmatrix}
\]

(17)

where \( \beta(i) = \lambda^{1/2} D^{-H}(i-1) \).

So, in order to downdate \( D^H(i) \) from \( D^H(i-1) \) we first compute the matrix \( P(i) = P_{L-1}(i) P_{L-2}(i) \cdots P_0(i) \) that implements an hyperbolic transform such that

\[
\begin{bmatrix}
    0_{L \times 1}^T \\
    1 / q^H(i)
\end{bmatrix} = \begin{bmatrix}
    -\beta(i) \\
    1
\end{bmatrix}
\]

(19)

and then, using \( P(i) \), we compute

\[
\begin{bmatrix}
    D^H(i) \\
    m^H(i)
\end{bmatrix} = \begin{bmatrix}
    P(i) \\
    0_{L \times 1}^T
\end{bmatrix} \begin{bmatrix}
    \lambda^{1/2} D^{-H}(i-1) \\
    0_{L \times 1}^T
\end{bmatrix}
\]

(20)

For details on construction of \( P(i) \) refer to Appendix.

Finally, the receiver filter for the desired user can be computed as

\[
w(i) = U^{-1}(i)U^{-H}(i)CD^{-1}(i)D^{-H}(i)p(i)
\]

(21)

Since in some applications such as blind user detection and beamforming the main interest is in the receiver filter output, \( y(i) = w^H(i)r(i) \), and not the receiver filter itself, some operations can be avoided. To do this, first note from (12) that

\[
D^H(i)D(i) = \lambda^{-1} D^H(i-1)D(i-1) - \lambda z(i)z^H(i)
\]

(22)

then, replacing (15) in (14) we get

\[
D(i) = \lambda^{-1/2} A(i)D(i-1) + v(i)z^H(i)
\]

(23)

Pre-multiplying (23) by \( D^H(i) \) and then equating to (22), we get

\[
v(i) = -D^{-H}(i)z(i)
\]

(24)
so that \( m(i) \) in (18) is given by

\[ m(i) = D^{-1}(i)D^{-H}(i)\mathbf{z}(i)/q(i) = \Omega^{-1}(i)\mathbf{z}(i)/q(i) \tag{25} \]

Now, by using a procedure similar to the one that led to (24) from (22) and (23), replacing (5) in (4) and after some algebraic manipulations, we get

\[ f(i) = U^{-H}(i)r(i) \tag{26} \]

that, when replaced in (8), yields

\[ \mathbf{z}(i) = -U^{-1}(i)U^{-H}(i)r(i)/\gamma(i) = -R^{-1}(i)r(i)/\gamma(i) \tag{27} \]

which is known as the Kalman gain.

Using (21), (27), (13) and (25), we have that the output signal can be computed directly as

\[ y(i) = w^{H}(i)r(i) \]

\[ = p^{H}(i)D^{-1}(i)D^{-H}(i)c^{H}(i)U^{-1}(i)U^{-H}(i)r(i) \]

\[ = -\gamma(i)p^{H}(i)D^{-1}(i)D^{-H}(i)c^{H}(i)\mathbf{z}(i) \]

\[ = -\gamma(i)p^{H}(i)D^{-1}(i)D^{-H}(i)\mathbf{x}(i) \]

\[ = -\gamma(i)q(i)p^{H}(i)m(i) \tag{28} \]

that are outputs of the CIQRD-RLS algorithm. The proposed CIQRD-RLS algorithm for linearly constrained minimum variance filtering is summarized in Table 1.

### 4. Simulation results

For simulation purposes we implement a minimum variance least-squares receiver for a downlink BPSK synchronous multicarrier (MC) CDMA transmission system with \( K \) users, as depicted in Fig. 2. For user \( k \), the transmitted symbols, \( b_k(i) \), drawn from a complex signal constellation with zero mean and unit average symbol energy, are first spread by a PN code \( c_k \) of \( M \) chips per symbol, \( M=32 \) in our simulations. The chips are grouped in blocks and transmitted in multicarrier fashion by a \( M \times M \) matrix \( F_k \), where \( F \) implements the normalized discrete Fourier transform.

In order to allow interblock interference (IBI) suppression at the receiver, a length \( L_g \) cyclic prefix (CP) insertion is performed before a transmission by a \( P \times M \) matrix \( T \), detailed below, where \( 0_{m \times n} \) represents an \( m \times n \) null matrix and \( P=M+L_g \); \( L_g \) must be at least the channel order to avoid IBI:

\[ T = \begin{bmatrix} 0_{L_g \times (M-L_g)} & I_{L_g} \\ I_M & \end{bmatrix} \]

Each block of chips is then serially transmitted through a multipath channel, modeled here as a FIR filter with \( L \) taps whose gains are samples, taken at the chip rate, of the channel impulse response complex envelope. It is assumed that during the \( i \)-th block duration the channel impulse response, \( h(i) = [h_0(i) \ldots h_{L-1}(i)]^T \), remains constant. In our simulations we set \( L=4 \) and \( L_g=3 \). Finally, through the use of a matrix \( G = [0_{M \times L_g} | I_M] \), the receiver removes the cyclic prefix from the received signal to eliminate IBI.

### Table 1

Proposed CIQRD-RLS algorithm.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| \( U(0) \) the Cholesky factor of \( R(0) = \delta \) \( \delta \) a small constant | \( D(0) \) the Cholesky factor of \( \mathbf{C}^{H}(i)R^{-1}(i)\mathbf{C}(i) \)

<table>
<thead>
<tr>
<th>1. Form the matrix–vector product:</th>
<th>( a(i) = \lambda^{-1/2}U^{-H}(i-1)r(i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Compute ( \mathbf{Q}(i) ) such that</td>
<td>( \begin{bmatrix} 1/\gamma(i) &amp; 0 \ 0_{M \times 1} &amp; -a(i) \end{bmatrix} )</td>
</tr>
<tr>
<td>3. Apply ( \mathbf{Q}(i) ) to</td>
<td>( \begin{bmatrix} \mathbf{z}^{H}(i) \ U^{-H}(i) \end{bmatrix} = \mathbf{Q}(i) \begin{bmatrix} \mathbf{0}<em>{M \times 1} \ 0</em>{L_g} \end{bmatrix} )</td>
</tr>
<tr>
<td>4. Form the matrix–vector product:</td>
<td>( \mathbf{x}(i) = \mathbf{C}^{H}(i)z(i) )</td>
</tr>
<tr>
<td>5. Compute ( \mathbf{P}(i) ) such that (see Appendix Table 2)</td>
<td>( \begin{bmatrix} 0_{L_g \times 1} \ 1/q^2(i) \end{bmatrix} = \mathbf{P}(i) \begin{bmatrix} -b(i) \ 1 \end{bmatrix} )</td>
</tr>
<tr>
<td>6. Apply ( \mathbf{P}(i) ) to ( \lambda^{-1/2}D^{-H}(i-1)0_{L_g} ) ( \mathbf{h}_k ) forming</td>
<td>( \begin{bmatrix} D^{-H}(i) \ m^{H}(i) \end{bmatrix} = \mathbf{P}(i) \begin{bmatrix} \lambda^{1/2}D^{-H}(i-1) \ \mathbf{0}_{L_g \times 1} \end{bmatrix} )</td>
</tr>
<tr>
<td>7. Compute the receiver filter output</td>
<td>( y(i) = -\gamma(i)q(i)p^{H}(i)m(i) )</td>
</tr>
<tr>
<td>8. Alternatively compute</td>
<td>( \begin{bmatrix} w(i) \end{bmatrix} = U^{-1}(i)U^{-H}(i)CD^{-1}(i)D^{-H}(i)p(i) )</td>
</tr>
</tbody>
</table>

After some algebraic manipulations, the \( M \)-dimensional observation vector, \( \mathbf{r}(i) \), can be represented by [24]

\[ \mathbf{r}(i) = \sum_{k=1}^{K} \sqrt{\rho_k} \mathbf{c}_k h(i)b_k(i) + \mathbf{n}(i) \tag{29} \]

where \( \rho_k \) is the average power of the transmitted symbol for user \( k \), \( \mathbf{n}(i) = [n_0(i) \ldots n_{M-1}(i)]^T \) is a complex white Gaussian noise vector with covariance matrix \( \mathbf{E}[\mathbf{n}(i)n^{H}(i)] = \sigma^2 I_M \) and \( \mathbf{c}_k \) is an \( M \times L_g \) code related circulant matrix for user \( k \), whose columns are circularly shifted versions of the \( k \)-th user transformed spreading sequence, \( F_k\mathbf{c}_k \).

Considering the observation vector in (29), the linearly constrained minimum variance least-squares receiver for user \( k \) is the filter \( \mathbf{w}_k(i) \) obtained as [24]

\[ \mathbf{w}_k(i) = \arg \min_{\mathbf{w}} \sum_{j=1}^{i} \lambda^{i-j} |\mathbf{w}^{H}\mathbf{r}(j)|^2 \quad \text{subject to} \quad \mathbf{c}_k^{H}\mathbf{w}(i) = \mathbf{p} \tag{30} \]

whose solution is

\[ \mathbf{w}_k(i) = R^{-1}(i)\mathbf{C}_k(c_k^H R^{-1}(i) c_k)^{-1} \mathbf{p} \tag{31} \]
where \( p \) is an estimate of the channel impulse response. In all the experiments vector \( p \) was made equal to the complex channel impulse response, \( h(t) \).

The performance of the blind receiver was measured in terms of the signal-to-interference plus noise ratio (SINR) and bit error rate (BER). We compare the proposed algorithm with the linearly constrained minimum variance fast recursive least-squares (LCMV FRLS) algorithm in \([17]\), the linearly constrained minimum variance IQRD-RLS (LCMV IQRD-RLS) algorithm in \([7]\) and the linearly constrained constant modulus IQRD-RLS receiver (LCCM IQRD-RLS) in \([6]\).

In the first experiment we set a system with \( K=10 \) users, the power level distribution amongst the interferers follow a log-normal distribution with associated standard deviation of 10 dB. The path gains of the transmission channel are drawn from a zero-mean and unit variance complex Gaussian random variable and kept fixed throughout each simulation run. The forgetting factor was set to \( \lambda = 0.999 \) and the results are an average of 100 runs. The obtained SINR performance results are depicted in Fig. 3 for \( E_b/N_0 = 0 \) dB, \( E_b/N_0 = 10 \) dB and \( E_b/N_0 = 20 \) dB with respect to the desired user power level (\( E_b \) is the energy per bit of the desired user).

Results indicate that the LCMV FRLS and the LCMV IQRD-RLS algorithms exhibit numerical instability as they diverge after a number of transmitted symbols, while the LCMV IQRD-RLS and the proposed algorithm are stable and presents similar steady-state performance for low signal-to-noise ratio (up to \( E_b/N_0 = 10 \) dB), with the proposed algorithm outperforming the LCMV IQRD-RLS in terms of rate of convergence. In Fig. 4 the obtained BER vs. \( E_b/N_0 \) for the same scenario is depicted. Results are an average of 100 independent runs of 2000 transmitted symbols.

Note that for \( E_b/N_0 > 10 \) dB the performance of the LCMV FRLS deteriorates because the ill-conditioned matrices involved in the algorithm. Results show that LCMV IQRD-RLS algorithm is not suitable for the implementation of a MC CDMA blind receiver, in fact, it was originally proposed for beamforming. Although the purpose of this paper is not to compare the bit error rate performance of a constrained constant modulus receiver (LCCM IQRD-RLS) with a constrained minimum variance receiver (Proposed) but their numerical stability, the results showed a superiority of the proposed algorithm over LCCM IQRD-RLS. This is also due to the fact that error counting includes the sequence of transmitted symbols as were the proposed algorithm performs better than LCMV IQRD-RLS algorithm in terms of SINR (the transient symbols).

In the second experiment we considered the BER performance in a dynamic scenario in which the system has initially seven users, the power level distribution amongst the interferers follow a log-normal distribution with associated standard deviation of 3 dB. After 1000 symbols, seven additional users enter the system and the power level distribution amongst interferes is loosen with associated standard deviation being increased to 10 dB. The path gains of the transmission channel are randomly drawn from a zero-mean and unit variance complex Gaussian random variable and kept fixed throughout each simulation run. The forgetting factor was set to \( \lambda = 0.999 \). The result, which is an average of 100 runs, is shown in Fig. 5 for \( E_b/N_0 = 20 \) dB. Note that the LCMV FRLS algorithm diverges after additional users enter to the system.

Finally, to evaluate the behavior of the algorithm for different values of forgetting factor, a MC CDMA transmission system in a time-variant random channel was simulated. The sequence of channel coefficients, \( h_l(i) = p_l z_l(i) \) \((l=0,1,2,\ldots,L−1)\) is obtained with Clarke’s model \([25]\). This procedure corresponds to the generation of \( L \) independent sequences of correlated unit power complex Gaussian random variables \( \{z_l^2(i)\} = 1 \) with the path weights \( p_l \) normalized so that \( \sum_{l=0}^{L} |p_l|^2 = 1 \). We assume that the average power of each path decays exponentially, such that \( |p_l(i)|^2 = \sigma_l^2 \exp{−l} \), \( l = 0,1,2,3 \), and \( \sigma_0^2 = 1−\exp{−1/(1−e^{-1})} \)[26]. The results, depicted in Fig. 6, are shown in terms of the normalized Doppler frequency \( f_d T \), where \( f_d \) is the
Doppler frequency and $T$ is the inverse of the transmission rate. In the simulations two values of $f_{DT}$ were assumed. For $f_{DT}=0.0001$ a value of $\lambda = 0.997$ was used, while for $f_{DT}=0.0005$, $\lambda = 0.996$. As expected, the BER of the two algorithms increase for a faster varying channel ($f_{DT}=0.0005$), but the proposed algorithm exhibits better bit error rate than the LCCM IQRD-RLS algorithm. Both algorithms behave numerically stable.

In Table 3 the computational complexity of the algorithms is given in terms of complex multiplications, complex sums and total number of operations for an hypothetic case of $M=32$ and $L=4$ (it was assumed that the computational
The cost of a complex multiplication and a complex sum is similar. The evaluation of computational complexity does not consider possible optimizations in the implementation. Results in Table 3 are valid for serial architectures. However, as mentioned in Introduction, QR decomposition based adaptive algorithms are suitable for systolic arrays or DSP vector architectures since they possess desirable properties for VLSI implementations such as regularity and good finite word-length behavior. Also they can be mapped to CORDIC arithmetic-based processors and can be designed efficiently for high-speed/low-power applications through pipelining and parallel processing. In serial implementations the system
clock must be several times faster than the sample rate due to the implementation of several matrix–vector multiplications, among other operations (see LCMV-FRLS algorithm [17] for example). On the other hand, inverse QR based algorithms can be implemented in parallel, such that the speed of the algorithm increases, at the expense of more CORDIC units (hardware complexity). In the proposed algorithm, two unconstrained IQR-RLS decomposition stages are used to update the LCMV filter coefficients, which can be efficiently implemented on CORDIC arithmetic-based processors. Between these two stages only one matrix–vector multiplication is found, see (13). However, (13) can also be efficiently computed in parallel, incurring only in an augmented hardware complexity and an increment of order $M$ in the latency. Other algorithms include several matrix–vector multiplication which forces a serial or pipelined implementation, and thus, reducing the speed of the algorithm.

In terms of stability, a floating-point implementation the LCMV FRLS algorithm, which uses the matrix inversion lemma to derive the algorithm to update $R^{-1}(i)$, produce errors between iterations and, in practice, the IQRD-RLS algorithm, which is known to be stable. The updating its Cholesky factor using the (unconstrained) downdating problem is not too ill-conditioned[4].

In [20] is shown that the downdating problem is ill-conditioned if any singular value is reduced significantly in the sense that acceptable results can always be expected as $M$ increases. In Table 3 is shown that the downdating problem is ill-conditioned if any singular value is reduced significantly (not necessarily becoming small). This happens in the presence of outliers, i.e., an erroneous and large element. However, in the proposed algorithm, the input for the downdating stage is the vector $x(i) = C^iR^{-1}(i)r(i)/\gamma(i)$ which is a filtered version of the input vector, so that we expect to not contain any outlier. Simulations results show good numerical properties of the proposed algorithm that corroborate the arguments above.

5. Conclusions

In this paper was proposed an IQRD-RLS algorithm for linearly constrained minimum variance filters. The proposed algorithm is suitable for implementations in systolic arrays or DSP vector architectures and, as shown through computer simulations, has better numerical behavior in finite precision environments when compared with other constrained IQRDRLS algorithms for blind detection in MC-CDMA systems. This numerical robustness results in better performance in terms of BER and SINR.

Appendix A. Construction of matrix $P(i)$

The structure of the matrix $P(i)$ is somewhat similar to the structure of the matrix $Q(i)$, and depends on the type of triangularization of the matrix $D(i)$. For example, if we consider that $D(i)$ is an upper triangular matrix, then the structure of $P(i)$ ($j = 0, \ldots, L-1$) is

$$P(i) = \begin{bmatrix} p_{j,00} & 0 & \cdots & 0 \\ 0 & I_j & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_{L-j} \end{bmatrix}$$

(A.1)

where the coefficients $p_{j,00}, j = 0,1$ are computed by an hyperbolic transform algorithm. In this paper we use the hyperbolic Householder transform, which proceed as follows. First define $\Phi = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, then, if we want to compute the matrix:

$$\hat{P}(i) = \begin{bmatrix} p_{j,00} & p_{j,01} \\ p_{j,10} & p_{j,11} \end{bmatrix}$$

(A.2)

such that

$$\begin{bmatrix} 0 \\ e(i) \end{bmatrix} = \hat{P}(i) \begin{bmatrix} \delta(i) \\ \kappa(i) \end{bmatrix}$$

(A.3)

where $|e(i)|^2 = |\kappa(i)|^2 - |\delta(i)|^2$ and $\hat{P}(i)\Phi\hat{P}(i) = \Phi$, we apply the algorithm in Table 2. More robust algorithms for hyperbolic Householder transforms can be found in the

Table 2: Hyperbolic householder transform [30].

<table>
<thead>
<tr>
<th>Define $\Phi = \begin{bmatrix} 1 \ 0 \end{bmatrix}$ and $a = \begin{bmatrix} 0 \ -\sqrt{-1} \end{bmatrix}$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Set $\eta(i) = [\delta(i) , \kappa(i)]^T$.</td>
</tr>
<tr>
<td>(2) Set $x(i) = \eta(i) + \sqrt{\eta^T(i)\Phi\eta(i)}a$.</td>
</tr>
<tr>
<td>(3) Set $v(i) = 2/(x^T(i)\Phi x(i))$.</td>
</tr>
<tr>
<td>(4) Compute $P(i)$ as $P(i) = I_2 - v(i)x(i)x^T(i)\Phi$.</td>
</tr>
</tbody>
</table>

Table 3: Computational complexity of RLS algorithms.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LCMV FRLS</td>
<td>$M^3 + M(2L^2 + 3L - 1) - L$</td>
<td>$2M^2 + M(2L^2 + 4L + 1) + 3L + 1$</td>
<td>6025</td>
</tr>
<tr>
<td>LCMV IQRD-RLS</td>
<td>$M^3 + M(4L + 2) - L - 2$</td>
<td>$M^2 + M(3L + 1) + L^2 + 2L + L + 1$</td>
<td>3135</td>
</tr>
<tr>
<td>LCMR IQRD-RLS</td>
<td>$M^3 + M(L^2 - 5L - 1) + 2L^2 - L + 2$</td>
<td>$M^2 + M(2L^2 + 5L + 4) + L^2 + L$</td>
<td>5010</td>
</tr>
<tr>
<td>Proposed</td>
<td>$2M^2 + M(2L - 3) + 2L^2 - 2L$</td>
<td>$2M^2 + 3ML + 2L + 2L + 2$</td>
<td>4706</td>
</tr>
</tbody>
</table>
literature [27,28], as well systolic implementations of the algorithms [29].

References