A NEW ORDER RECURSIVE MULTIPLE ORDER MULTICHANNEL FAST QRD-RLS ALGORITHM

António L. L. Ramos †, José A. Apolinário Jr. †, and Marcio G. Siqueira‡

†IME–DE/3
Praça General Tibúrcio, 80
22290-270, Rio de Janeiro, RJ, Brazil
antonioluis@ime.eb.br and apolin@ieee.org

‡Cisco Systems
170 West Tasman Drive
San Jose, CA 95134–1706 USA
mgs@cisco.com

ABSTRACT

In many adaptive filtering applications, such as in the case of the Volterra filters, the use of channels of unequal orders are common. This paper introduces a new Multichannel Fast QRD-RLS algorithm based on the updating of *a posteriori* backward errors that attains both cases of channels of equal or unequal orders. This new algorithm exhibits good numerical behavior and is order recursive, which allows a systolic array implementation with lower computational complexity compared to earlier proposed algorithms.

1. INTRODUCTION

Digital processing of multichannel signals using adaptive filters has recently found a variety of new applications including color image processing, multi-spectral remote sensing imagery, biomedicine, channel equalization, stereophonic echo cancellation, multidimensional signal processing, Volterra–type nonlinear system identification, and speech enhancement [1]. This increased number of applications has spawned a renewed interest in efficient multichannel algorithms. One class of algorithms, known as multichannel Fast QRD-RLS adaptive algorithms based on backward errors updating, *a priori* [2] or *a posteriori* [3], has become an attractive option because of their fast convergence and reduced computational complexity.

Unified formulations of Fast QRD-RLS algorithms are available in [4], for the single channel case, and in [5], for the multichannel case. In this paper, a new order recursive multiple order Multichannel Fast QRD-RLS algorithm is developed based upon fixed and multiple order multichannel algorithms recently proposed in [6] and [7], using an approach similar to [3] and [6]. This new order recursive multichannel Fast QRD algorithm, using the *a posteriori* backward prediction errors updating, can be used in problems with channels of equal or unequal orders while comprising only scalar operations. The new algorithm is particularly suitable for systolic array implementation.

The QRD-RLS family of Multichannel algorithms uses the weighted least-squares (LS) objective function defined as

$$\xi_{LS}(k) = \sum_{i=0}^{k} \lambda^{k-i} e^2(i) = \boldsymbol{e}^T(k)\boldsymbol{e}(k) \tag{1}$$

The authors thank CNPq, CAPES, and FAPERJ for partial funding of this paper.

where vector $\boldsymbol{e}(k) = \begin{bmatrix} e(k) & \lambda^{1/2}e(k-1) & \cdots & \lambda^{k/2}e(0) \end{bmatrix}^T$ can be represented as follows.

$$\begin{aligned}\boldsymbol{e}(k) &= \begin{bmatrix} d(k) \\ \lambda^{1/2}d(k-1) \\ \vdots \\ \lambda^{k/2}d(0) \end{bmatrix} - \begin{bmatrix} \boldsymbol{x}_N^T(k) \\ \lambda^{1/2}\boldsymbol{x}_N^T(k-1) \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_N^T(0) \end{bmatrix} \boldsymbol{w}_N(k) \\ &= \boldsymbol{d}(k) - \boldsymbol{X}_N(k)\boldsymbol{w}_N(k) \end{aligned} \tag{2}$$

where

$$\boldsymbol{x}_N^T(k) = \begin{bmatrix} \boldsymbol{x}_k^T & \boldsymbol{x}_{k-1}^T & \cdots & \boldsymbol{x}_{k-N+1}^T \end{bmatrix} \tag{3}$$

and $\boldsymbol{x}_k^T = [x_1(k) \quad x_2(k) \quad \cdots \quad x_M(k)]$ is the input signal vector at instant k. Note that N is initially defined as the number of filter coefficients per channel (fixed order), M is the number of input channels, and $\boldsymbol{w}_N(k)$ is the $MN \times 1$ coefficient vector at time instant k.

If $\boldsymbol{U}_N(k)$ stands for the Cholesky factor of $\boldsymbol{X}_N^T(k)\boldsymbol{X}_N(k)$, obtained through the Givens rotation matrix $\boldsymbol{Q}_N(k)$, then

$$\begin{aligned}\boldsymbol{e}_q(k) = \boldsymbol{Q}_N(k)\boldsymbol{e}(k) &= \begin{bmatrix} \boldsymbol{e}_{q1}(k) \\ \boldsymbol{e}_{q2}(k) \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{d}_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} - \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}_N(k) \end{bmatrix} \boldsymbol{w}_N(k) \end{aligned} \tag{4}$$

and the optimal coefficient vector, $\boldsymbol{w}_N(k)$, is obtained by making $\boldsymbol{e}_{q2}(k)$ a null vector. In order to cope with the multiple order case, the input signal vector needs to be redefined.

This paper is organized as follows. In Section 2, a new order recursive multiple-order Multichannel Fast QRD based on the updating of the *a posteriori* error vector is introduced. Simulation results and conclusions are Summarized in Sections 3 and 4, respectively.

2. THE NEW MULTIPLE ORDER MULTICHANNEL FAST QRD-RLS ALGORITHM

The starting point for the derivation of multiple order multichannel fast QRD-RLS algorithms [3], [6], and [7] is the construction of the input vector such that the general case of equal or unequal order is attained. It is also taken into account the fact that M steps are executed for each iteration of the algorithm. This means that the

M channels are processed separately; yet they are interdependent: the quantities collected after the *i*-th channel is processed are used as initial values for the processing of the (*i+1*)-th channel and so on. Finally, the quantities collected during the processing of the last channel, in a given instant k, are used as initial values for the processing of the first channel in $k+1$. This will become clear during the derivation of the algorithm.

The following notation is adopted:

M	is the number of input channels;
$N_1, N_2, \cdots, N_M$	are the number of *taps* for each input channel;
$N = \sum_{r=1}^{M} N_r$	Overall number of *taps*.

Without loss of generality, it is assumed here that $N_1 \geq N_2 \geq \cdots \geq N_M$.

2.1. Redefining the input vector

From the fact that the M channels are processed separately, the updating of the input vector is performed likewise: in a given time instant k, we have vector $\boldsymbol{x}_N(k)$[1] from which we successively obtain $\boldsymbol{x}_{N+1}(k+1)$ by appending the most recent sample of channel one at time instant $k+1$, $\boldsymbol{x}_{N+2}(k+1)$ by appending the most recent sample of channel two, and so on. At the end of this process, we have the updated vector $\boldsymbol{x}_{N+M}(k+1)$.

Actually, this is not that simple because the position to be occupied by the newer samples from each channel in the updated vector $\boldsymbol{x}_{N+M}(k+1)$ must be carefully determined. The vector $\boldsymbol{x}_N(k)$, used as the starting point to obtain $\boldsymbol{x}_{N+M}(k+1)$ is construct as follows: we first choose $N_1 - N_2$ *samples* from the first channel to be the leading elements of $\boldsymbol{x}_N(k)$, followed by $N_2 - N_3$ *pairs of samples* from the first and second channels, followed by $N_3 - N_4$ *triples of samples* of the first three channels and so far till the $N_M - N_{M+1}$ *M-ples of samples* of all channels. It is assumed that $N_{M+1} = 0$. The position p_i of the most recent sample of the *i*-th channel can be expressed compactly as [3] $p_i = \sum_{r=1}^{i-1} r(N_r - N_{r+1}) + i$, for $i = 1, 2, \cdots, M$. Moreover, the M successive input vectors for a given instant k, obtained from $\boldsymbol{x}_N(k)$, can be defined as follows.

$$\boldsymbol{x}_{N+1}^T(k+1) = \begin{bmatrix} x_1(k+1) & \boldsymbol{x}_N^T(k) \end{bmatrix} \tag{5}$$

$$\boldsymbol{x}_{N+i}^T(k+1) = \begin{bmatrix} x_i(k+1) & \boldsymbol{x}_{N+1-i}^T(k+1) \end{bmatrix} \boldsymbol{P}_i \tag{6}$$

where $\boldsymbol{P}_i$ is a permutation matrix which takes the most recent sample $x_i(k+1)$ of the *i*-th channel to the position p_i, after left shifting the first $p_i - 1$ elements of $\boldsymbol{x}_{N-i+1}^T(k+1)$. After concluding this process for the M channels, it can be observed that $\boldsymbol{x}_{N+M}^T(k+1) = [\boldsymbol{x}_N^T(k+1) \quad x_1(k-N_1+1) \cdots \quad x_M(k-N_M+1)]$ which clearly means that the first N elements of $x_{N+M}^T(k+1)$ provide the input vector of the next iteration. We can now define the input data matrices as follows.

$$\boldsymbol{X}_{N+i}(k) = \begin{bmatrix} \boldsymbol{x}_{N+i}^T(k) \\ \lambda^{1/2}\boldsymbol{x}_{N+i}^T(k-1) \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_{N+i}^T(0) \end{bmatrix} \quad i = 1, 2, \cdots, M \tag{7}$$

Let $\boldsymbol{U}_{N+i}(k)$ be the Cholesky factor of $\boldsymbol{X}_{N+i}^T(k)\boldsymbol{X}_{N+i}(k)$; we can define the *a posteriori* backward error vector, $\boldsymbol{f}_{N+i}(k+1)$, as

$$\boldsymbol{f}_{N+i}(k+1) = \boldsymbol{U}_{N+i}^{-T}(k+1)\boldsymbol{x}_{N+i}(k+1) \quad \textit{for } i = 1, 2, \cdots, M. \tag{8}$$

From (8) and the definition of the input vector, we can write

$$\boldsymbol{f}_{N+M}(k+1) = \begin{bmatrix} \boldsymbol{f}_N(k+1) \\ \boldsymbol{f}^{(N)}(k+1) \end{bmatrix} \tag{9}$$

where $\boldsymbol{f}^{(N)}(k+1)$ are the last M elements of $\boldsymbol{f}_{N+M}(k+1)$.

The updating of $\boldsymbol{f}_{N+i}(k+1)$ is accomplished in M forward steps at each instant k:
$\boldsymbol{f}_N(k) \rightarrow \boldsymbol{f}_{N+1}(k+1) \rightarrow \boldsymbol{f}_{N+2}(k+1) \rightarrow \cdots \cdots \rightarrow \boldsymbol{f}_{N+M}(k+1)$

2.2. Triangularization of the information matrix

Equation (7) suggests that the updating of the information matrix is performed in M forward steps for each iteration.

2.2.1. First step ($i = 1$):

$\boldsymbol{X}_{N+1}(k)$ can be defined as

$$\boldsymbol{X}_{N+1}(k) = \left[\begin{array}{c|c} \boldsymbol{d}_f^{(1)}(k) & \begin{array}{c} \boldsymbol{X}_N(k-1) \\ \boldsymbol{0}^T \end{array} \end{array} \right] \tag{10}$$

where $\boldsymbol{d}_{f1}^{(1)}(k) = [x_1(k) \quad \lambda^{1/2}x_1(k-1) \quad \cdots \quad \lambda^{k/2}x_1(0)]$.

If $\boldsymbol{U}_N(k-1)$ and $\boldsymbol{Q}_N^{(1)}(k)$ stand, respectively, for the Cholesky factor of $\boldsymbol{X}_N^T(k-1)\boldsymbol{X}_N(k-1)$ and the orthogonal matrix associated to this process, we can write, from (10), that

$$\begin{bmatrix} \boldsymbol{Q}_N^{(1)}(k) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{1\times 1} \end{bmatrix} \left[\begin{array}{c|c} \boldsymbol{d}_f^{(1)}(k) & \begin{array}{c} \boldsymbol{X}_N(k-1) \\ \boldsymbol{0}^T \end{array} \end{array} \right] = \begin{bmatrix} \boldsymbol{e}_{fq1}^{(1)}(k) & \boldsymbol{0} \\ \boldsymbol{d}_{fq2}^{(1)}(k) & \boldsymbol{U}_N(k-1) \\ \lambda^{k/2}x_1(0) & \boldsymbol{0}^T \end{bmatrix} \tag{11}$$

To complete the triangularization process of $\boldsymbol{X}_{N+1}(k)$ leading to $\boldsymbol{U}_{N+1}(k)$, we premultiply (11) by two other Given rotation matrices as follows.

$$\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}_{N+1}(k) \end{bmatrix} = \boldsymbol{Q'}_f^{(1)}(k)\boldsymbol{Q}_f^{(1)}(k)\boldsymbol{X}_{N+1}(k) \tag{12}$$

where $\boldsymbol{Q}_f^{(1)}(k)$ is the orthogonal matrix responsible for zeroing the first, $k-N$ rows, generating $e_{fN}^{(1)}(k)$ as in (13), and $\boldsymbol{Q'}_f^{(1)}(k)$ completes the triangularization process by zeroing $\boldsymbol{d}_{fq2}^{(1)}(k)$ from (12) in a top down procedure against $e_{fN}^{(1)}(k)$. After removing the resulting null section in the upper part of (12), the result is:

$$\boldsymbol{U}_{N+1}(k) = \boldsymbol{Q'}_{\theta f}^{(1)}(k) \begin{bmatrix} \boldsymbol{d}_{fq2}^{(1)}(k) & \boldsymbol{U}_N(k-1) \\ e_{fN}^{(1)}(k) & \boldsymbol{0}^T \end{bmatrix} \tag{13}$$

From (13), we obtain the following relation that will be useful to obtain an expression for the updating of $\boldsymbol{f}_N(k)$.

$$[\boldsymbol{U}_{N+1}(k+1)]^{-1} = \begin{bmatrix} \boldsymbol{0}^T & \frac{1}{e_{fN}^{(1)}(k+1)} \\ \boldsymbol{U}_N^{-1}(k) & -\frac{1}{e_{fN}^{(1)}(k+1)}\boldsymbol{U}_N^{-1}(k)\boldsymbol{d}_{fq2}^{(1)}(k+1) \end{bmatrix} \cdot \left[\boldsymbol{Q'}_{\theta f}^{(1)}(k+1)\right]^T \tag{14}$$

[1] The subscript $_N$ denotes the N-th order vector.

From (13), we know that ${\boldsymbol{Q}'_{\theta f}}^{(1)}(k)$ is the Givens rotation matrix responsible for zeroing $\boldsymbol{d}^{(1)}_{fq2}(k)$ against $e^{(1)}_{fN}(k)$. Thus, it is straightforward to write

$$\begin{bmatrix} \boldsymbol{0} \\ e^{(1)}_{f0}(k+1) \end{bmatrix} = {\boldsymbol{Q}'_{\theta f}}^{(1)}(k+1) \begin{bmatrix} \boldsymbol{d}^{(1)}_{fq2}(k+1) \\ e^{(1)}_{fN}(k+1) \end{bmatrix} \quad (15)$$

Now, recalling (8), we can use (14) and (5) to obtain a recursive expression to update $\boldsymbol{f}_{N+1}(k+1)$.

$$\boldsymbol{f}_{N+1}(k+1) = {\boldsymbol{Q}'_{\theta f}}^{(1)}(k+1) \begin{bmatrix} \boldsymbol{f}_N(k) \\ p^{(1)}(k+1) \end{bmatrix} \quad (16)$$

where

$$p^{(1)}(k+1) \quad = \quad \frac{{e_N}^{(1)}(k+1)}{e^{(1)}_{fN}(k+1)} \quad (17)$$

with ${e_N}^{(1)}(k+1)$ being the *a posteriori* error of the forward prediction for the first channel and $|e^{(1)}_{f_N}(k+1)|$ is given by the following expression.

$$|e^{(1)}_{f_N}(k+1)| = \sqrt{\left(\lambda^{1/2}|e^{(1)}_{f_N}(k)|\right)^2 + |(e^{(i)}_{fq1_N}(k+1)|^2} \quad (18)$$

The updating of $\boldsymbol{d}^{(1)}_{fq2}(k)$ is performed according to

$$\begin{bmatrix} \tilde{e}^{(1)}_{fq1}(k+1) \\ \boldsymbol{d}^{(1)}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}^{(0)}_{\theta_N}(k) \begin{bmatrix} x_1(k+1) \\ \lambda^{1/2}\boldsymbol{d}^{(1)}_{fq2}(k) \end{bmatrix} \quad (19)$$

and matrix $\boldsymbol{Q}^{(1)}_{\theta_{N+1}}(k+1)$, necessary in the next steps, is obtained from

$$\boldsymbol{Q}^{(1)}_{\theta_{N+1}}(k+1) \begin{bmatrix} 1 \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \gamma^{(i)}_{N+1}(k+1) \\ \boldsymbol{f}_{N+1}(k+1) \end{bmatrix} \quad (20)$$

2.2.2. *Following steps ($i > 1$):*

The input information matrix $\boldsymbol{X}_{N+i}(k)$ is related to $\boldsymbol{X}_{N+i-1}(k)$ according to

$$\boldsymbol{X}_{N+i}(k) = \begin{bmatrix} x_i(k) & \\ \lambda^{1/2}x_i(k-1) & \\ \vdots & \boldsymbol{X}_{N+i-1}(k) \\ \lambda^{k/2}x_i(0) & \end{bmatrix} \boldsymbol{P}_i \quad (21)$$

As in the first step, $\boldsymbol{X}_{N+i}(k)$ must be triangularized generating $\boldsymbol{U}_{N+i}(k)$ that corresponds to the Cholesky factor of $\boldsymbol{X}^T_{N+i}(k)\boldsymbol{X}_{N+i}(k)$. This process is detailed as follows. If $\boldsymbol{Q}_{\theta_{N+1-i}}(k)$ stands for orthogonal matrix obtained from the QR decomposition of $\boldsymbol{X}_{N+i-1}(k)$, we can write from (21)

$$\begin{bmatrix} \boldsymbol{Q}_{\theta_{N+1-i}}(k) & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{X}_{N+1-i}(k) \\ \boldsymbol{0}^T \end{bmatrix}$$
$$= \begin{bmatrix} \boldsymbol{e}^{(i)}_{fq1_{N+1-i}}(k) & \boldsymbol{0} \\ \boldsymbol{d}^{(i)}_{fq2}(k) & \boldsymbol{U}_{N+i-1}(k) \\ \boldsymbol{0} & \boldsymbol{0}^T \end{bmatrix} \boldsymbol{P}_i \quad (22)$$

Equation (22) results from the annihilation of $\boldsymbol{e}^{(i)}_{fq1_{N+1-i}}(k)$ against the first element of the last row of the matrix, using an appropriate orthogonal factor, and removing the resulting null section.

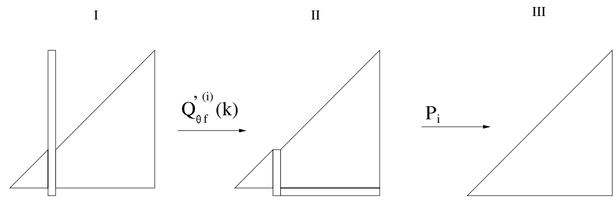

Fig. 1. Obtaining the lower triangular factor $\boldsymbol{U}_{N+i-1}(k)$.

The existence of the permutation matrix $\boldsymbol{P}_i$ in (22) prevents a direct annihilating of $\boldsymbol{d}^{(i)}_{fq2}(k)$ against $e^{(i)}_{f_{N+i-1}}(k)$ to complete the triangularization of matrix $\boldsymbol{X}_{N+1-i}(k)$. Fig. 1 illustrates the application of the Givens rotations under these circumstances. This process can be summarized as follows. The permutation factor, $\boldsymbol{P}_i$, right shifts $\boldsymbol{d}^{(i)}_{fq2}(k)$ to the i-th position as shown in the first part of the figure. Afterwards, a set of $N+i-p_i$ Given rotation matrices, ${\boldsymbol{Q}'_{\theta f}}^{(i)}$, are used to nullify the first $N+i-p_i$ elements of $\boldsymbol{d}^{(i)}_{fq2}(k)$ against $e^{(i)}_{f_{N+i-1}}(k)$ in a top down procedure. In order to obtain the desired triangular structure, we need another permutation factor that moves the last row of the matrix to the $N-p_i+1$ position, after downshifting the previous $N-p_i$ rows. This permutation factor coincides with $\boldsymbol{P}_i$.

The positive definiteness of the lower triangular matrix $\boldsymbol{U}_{N+i-1}(k)$, obtained as described above, is guaranteed if its diagonal elements, along with $e^{(i)}_{f_{N+i-1}}(k)$, are positive. Recalling that $e^{(i)}_{f_{N+i-1}}(k)$ is, actually, the absolute value of the forward error, the latter is valid and, $\boldsymbol{U}_{N+i-1}(k)$ being properly initialized, its positive definiteness is guaranteed. The procedure above can be written compactly as

$$\begin{aligned} \boldsymbol{U}_{N+i}(k) \quad &= \quad \boldsymbol{P}_i {\boldsymbol{Q}'_{\theta f}}^{(i)}(k) \\ &\cdot \begin{bmatrix} \boldsymbol{d}^{(i)}_{fq2}(k) & \boldsymbol{U}_{N+i-1}(k) \\ \boldsymbol{e}^{(i)}_{f_{N+i-1}}(k) & \boldsymbol{0}^T \end{bmatrix} \boldsymbol{P}_i \end{aligned} \quad (23)$$

From (23), the following relation can be derived.

$$\begin{aligned} [\boldsymbol{U}_{N+i}(k+1)]^{-1} \quad &= \quad \boldsymbol{P}^T_i \\ &\cdot \begin{bmatrix} \boldsymbol{0}^T & \frac{1}{e^{(i)}_{f_{N+i-1}}(k+1)} \\ \boldsymbol{U}^{-1}_{N+i-1}(k+1) & -\frac{\boldsymbol{U}^{-1}_{N+i-1}(k+1)\boldsymbol{d}^{(i)}_{fq2}(k+1)}{e^{(i)}_{f_{N+i-1}}(k+1)} \end{bmatrix} \\ &\cdot {\boldsymbol{Q}'^{T}_{\theta f}}^{(i)}(k+1)\boldsymbol{P}^T_i \end{aligned} \quad (24)$$

From (24), (6), and (8), it is possible to obtain the following recursive expression to compute $\boldsymbol{f}_{N+i}(k+1)$.

$$\boldsymbol{f}_{N+i}(k+1) \; = \; \boldsymbol{P}_i {\boldsymbol{Q}'_{\theta f}}^{(i)}(k+1) \begin{bmatrix} \boldsymbol{f}_{N+i-1}(k+1) \\ p^{(i)}_{N+i-1}(k+1) \end{bmatrix} \quad (25)$$

where

$$p^{(i)}_{N+i-1}(k+1) \quad = \quad \frac{e^{(i)}_{N+i-1}(k+1)}{e^{(i)}_{f_{N+i-1}}(k+1)} \quad (26)$$

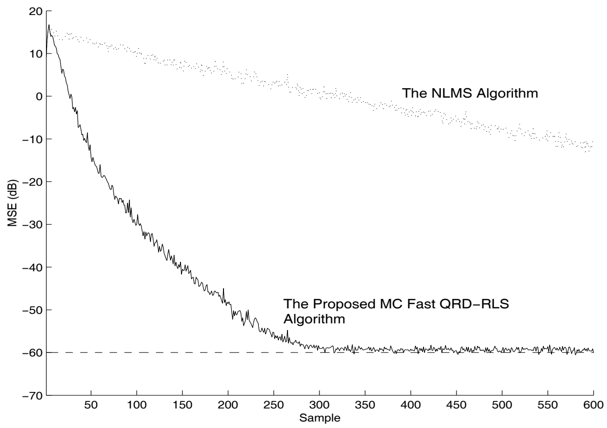

Fig. 2. Learning curves.

The scalar quantity $e^{(i)}_{N+i-1}(k+1)$ is the *a posteriori* forward prediction error for the *i*-th channel and $|e^{(i)}_{f_{N+i-1}}(k+1)|$ is given by the expression below.

$$|e^{(i)}_{f_{N+i-1}}(k+1)| = \sqrt{\left(\lambda^{1/2}|e^{(i)}_{f_{N+i-1}}(k)|\right)^2 + |e^{(i)}_{fq1_{N+i-1}}(k+1)|^2} \quad (27)$$

Now, looking carefully at (25) and recalling the definitions of $\boldsymbol{P}_i$ and ${\boldsymbol{Q}'}_{\theta f}^{(i)}(k+1)$, we conclude that the last $p_i - 1$ elements of $\boldsymbol{f}_{N+i}(k+1)$ and $\boldsymbol{f}_{N+i-1}(k+1)$ are identical. This is an important observation that allows a significant reduction in the computational complexity of the algorithm.

The updating of $\boldsymbol{d}^{(i)}_{fq2}(k)$ is performed according to

$$\begin{bmatrix} \tilde{e}^{(i)}_{fq1}(k+1) \\ \boldsymbol{d}^{(i)}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}^{(i-1)}_{\theta_{N+i-1}}(k+1) \begin{bmatrix} x_i(k+1) \\ \lambda^{1/2}\boldsymbol{d}^{(i)}_{fq2}(k) \end{bmatrix} \quad (28)$$

and the Givens rotations matrices $\boldsymbol{Q}_{\theta_{N+i}}(k+1)$, needed in the next forward step, are obtained as follows.

$$\boldsymbol{Q}^{(i)}_{\theta_{N+i}}(k+1) \begin{bmatrix} 1 \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \gamma^{(i)}_{N+i}(k+1) \\ \boldsymbol{f}_{N+i}(k+1) \end{bmatrix} \quad (29)$$

After the main loop ($i = 1 : M$), i.e., after the *M-th* channel is processed, the join process estimation is performed according to

$$\begin{bmatrix} e_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}^{(0)}_{\theta}(k+1) \begin{bmatrix} e_{q1}(k) \\ \boldsymbol{d}_{q2}(k) \end{bmatrix} \quad (30)$$

In order to achieve an order recursive structure to the algorithm, $p^{(i)}_j(k+1)$ and $|e^{(i)}_{f_j}(k+1)|$ must be expressed as in the two following equations.

$$|e^{(i)}_{f_j}(k+1)| = \sqrt{\left(\lambda^{1/2}|e^{(i)}_{f_j}(k)|\right)^2 + \left(e^{(i)}_{fq1_j}(k+1)\right)^2}$$
$$i = 1, 2, \cdots, M \qquad j = p_i, \cdots, L \quad (31)$$

and

$$p^{(i)}_j(k+1) = \frac{\gamma^{(i-1)}_j(k)e^{(i)}_{fq1_j}(k+1)}{|e^{(i)}_{f_j}(k+1)|}$$
$$j = p_i, \cdots, L \qquad i = 1, 2, \cdots, M \quad (32)$$

Table 1. Computational complexity (complex environment.)*

Algorithm	Mults.	Divs.	Sqrts
[7] †	$14NM + 13M$ $-9\sum_{i=1}^{M} p_i + 5N$	$3NM + 4M$ $-3\sum_{i=1}^{M} p_i$	$2NM + 3M$ $-2\sum_{i=1}^{M} p_i$
[3] †	$15NM + 14M$ $-10\sum_{i=1}^{M} p_i + 5N$	$4NM + 5M$ $-4\sum_{i=1}^{M} p_i$	$2NM + 3M$ $-2\sum_{i=1}^{M} p_i$
Table 2 ‡	$14NM + 13M$ $-9\sum_{i=1}^{M} p_i + 5N$	$4NM + 5M$ $-4\sum_{i=1}^{M} p_i$	$2NM + 3M$ $-2\sum_{i=1}^{M} p_i$
[3] ‡	$15NM + 14M$ $-10\sum_{i=1}^{M} p_i + 5N$	$5NM + 6M$ $-5\sum_{i=1}^{M} p_i$	$2NM + 3M$ $-2\sum_{i=1}^{M} p_i$

* Note that $p_i = \sum_{r=1}^{i-1} r(N_r - N_{r+1}) + i, \quad i = 1, 2, \cdots, M.$
† *Direct Form*, ‡ *Recursive Form.*

The computational complexity of the proposed algorithm is shown in Table 1 and the complete order recursive algorithm, for a the general case of a complex implementation, is summarized in Table 2.

3. SIMULATION RESULTS

The performance of the proposed algorithm is evaluated in a nonlinear system identification. The plant is a truncated second order Volterra system [8] which can be described as

$$d(k) = \sum_{n_1=0}^{L-1} w_{n_1}(k)x(k-n_1) + \sum_{n_1=0}^{L-1}\sum_{n_2=0}^{L-1} w_{n_1,n_2}(k)x(k-n_1)x(k-n_2) + \rho(k) \quad (33)$$

Equation (33) can be easily reformulated as a multichannel problem with $M = L + 1$ channels, where the most recent sample of the *i*-th channel is

$$x_i(k) = \begin{cases} x(k), & i = 1 \\ x(k)x(k-i+2), & i = 2, \cdots, L+1 \end{cases}$$

and the *i*-th channel order is

$$N_i = \begin{cases} L, & i = 1, 2 \\ L - i + 2, & i = 3, \cdots, L+1. \end{cases}$$

In our experiment, we used $L = 4$, $\lambda = 0.98$, and an SNR of $60dB$. The learning curve for the proposed algorithm is compared with the Normalized LMS (NLMS) [9] and the Inverse QRD-RLS [10] algorithms in Fig. 2. Its convergence speed is shown to be the same of the multiple order algorithms proposed in [3] (direct and order recursive versions) and in [7] (direct version). When compared to the NLMS and the IQRD-RLS algorithms, these algorithms have shown a better convergence speed performance. In terms of computational complexity, we have the following scenario. The algorithm proposed in this work, as can be seen from Table 1, has a lower computational complexity (multiplications and divisions) if compared to those (*direct* and *recursive forms*) based on backward prediction errors updating in [3]. But, also from Table 1, it can be seen that its order recursiveness has a cost in terms of the computational complexity when compared to its direct form counterpart of [7]. This new algorithm has proven to be stable and robust as expected in algorithms that use numerically stable Givens rotations to perform QR decomposition.

4. CONCLUSIONS

Block multichannel RLS-based algorithms are well known for their high computational complexity. Among these algorithms, the FQRD-RLS algorithms based on backward errors [2, 3, 7] are attractive due to their reduced computational complexity. The Multichannel FRQD-RLS algorithm based on *a priori* errors using the alternative input vector formulation described in Section II was shown to be an attractive option for the case of multiple order multichannel applications. This paper introduced the order recursive multichannel FQRD-RLS algorithm based on *a posteriori* error updating. This new algorithm exhibits the lowest complexity among known recursive order multichannel FQRD-RLS algorithms, while keeping all desirable numerical properties of its family.

Table 2. The New Order Recursive Multiple Order Multichannel Fast QRD-RLS Algorithm (complex version)

Initializations:

$\boldsymbol{d}^{(i)}_{fq2} = zeros(N,1); \quad \boldsymbol{f}^{(M)}(0) = \boldsymbol{0}; \quad \boldsymbol{d}_{q2} = \boldsymbol{0};$
$\gamma_N^{(0)}(0) = 1; \quad e^{(i)}_{f_N}(0) = \mu; \quad i = 1, 2, \cdots, M.$
All cosines = 1; all sines = 0;

for $k = 1, 2, \cdots$
{ $\gamma_0^{(1)} = 1; \quad e^{(0)}_{q1}(k+1) = d^*(k+1);$
$|e_0^{(1)}(k+1)| = \sqrt{\left(\lambda^{1/2}|e_0^{(1)}(k)|\right)^2 + |x_1(k+1)|^2};$
$\boldsymbol{f}^{(1)}_{N+1}(k+1) = [x_1(k+1)]^* / |e_0^{(1)}(k+1)|;$
for $i = 1 : M$,
{ $e^{(i)}_{fq1_0}(k+1) = x_i(k+1)$
for $j = 1 : N$,
{ $e^{(i)}_{fq1_j}(k+1) = \cos\left[\theta_j^{(i-1)}(k)\right] e^{(i)}_{fq1_{j-1}}(k+1) + \lambda^{1/2} \sin\left[\theta_j^{(i-1)}(k)\right] \boldsymbol{d}^{(i)}_{fq2_{N-j+1}}(k);$
$\boldsymbol{d}^{(i)}_{fq2_{N-j+1}}(k) = \lambda^{1/2} \cos\left[\theta_j^{(i-1)}(k)\right] \boldsymbol{d}^{(i)}_{fq2_{N-j+1}}(k) - \sin\left[\theta_j^{(i-1)}(k)\right]^* e^{(i)}_{fq1_{j-1}}(k+1);$
if $j \geq p_i - 1$,
$|e^{(i)}_{f_j}(k+1)| = \sqrt{\left(\lambda^{1/2}|e^{(i)}_{f_j}(k)|\right)^2 + |e^{(i)}_{fq1_N}(k+1)|^2};$
$p_j^{(i)}(k+1) = \dfrac{\gamma_j^{(i-1)}(k)\left[e^{(i)}_{fq1_j}(k+1)\right]^*}{|e^{(i)}_{f_j}(k+1)|};$
if $j = p_i - 1$,
$\boldsymbol{f}^{(i)}_{N+1-j+1}(k+1) = p_j^{(i)}(k+1);$
if $j > p_i - 1$,
$\cos\theta'^{(i)}_{f_j}(k+1) = |e^{(i)}_{f_j}(k+1)| / |e^{(i)}_{f_{j-1}}(k+1)|;$
$\sin\theta'^{(i)}_{f_j}(k+1) = \left[\cos\theta'^{(i)}_{f_j}(k+1) \cdot \boldsymbol{d}^{(i)}_{fq2_{N-j+1}}(k+1) / e^{(i)}_{f_j}(k+1)\right]^*;$
$\boldsymbol{f}^{(i)}_{N-j+1}(k+1) = \cos\theta'^{(i)}_{f_j}(k+1) \boldsymbol{f}^{(i-1)}_{N-j+2}(k+1) - \sin\left[\theta'^{(i)}_{f_j}(k+1)\right]^* p_j^{(i)}(k+1);$
$\sin\theta_j^{(i)}(k) = -\left[\boldsymbol{f}^{(i)}_{N-j+2}(k+1)\right]^* / \gamma^{(i)}_{j-1};$
$\cos\theta_j^{(i)}(k) = \sqrt{1 - |\sin\theta_j^{(i)}(k)|^2};$
$\gamma_j^{(i)}(k) = \cos\theta_j^{(i)}(k) \gamma^{(i)}_{j-1}(k+1);$
} for j
} for i
for $j = 1 : N$ % Join process estimation:
{ $e^{(j)}_{q1}(k+1) = \cos\theta_j^{(0)}(k+1) \boldsymbol{e}^{(j-1)}_{q1}(k+1) + \lambda^{1/2} \sin\theta_j^{(0)}(k+1) \boldsymbol{d}^{(N-j+1)}_{q2}(k);$
$\boldsymbol{d}^{(N-j+1)}_{q2}(k+1) = \lambda^{1/2} \cos\theta_j^{(0)}(k+1) \boldsymbol{d}^{(N-j+1)}_{q2}(k) - \sin\left[\theta_j^{(0)}(k+1)\right]^* e^{(j-1)}_{q1}(k+1);$
}
$e(k+1) = \left[e^{(N)}_{q1}(k+1)\right]^* / \gamma_N^{(0)}(k+1);$ % the *a priori* error
} for k

Obs.: The *asterisc* $(*)$ denotes complex conjugation.
$\theta_j^{(M)}(k) = \theta_j^{(0)}(k+1)$ and $\boldsymbol{f}^{(M)}_{N-j+2}(k) = \boldsymbol{f}^{(0)}_{N-j+2}(k+1)$.

5. REFERENCES

[1] N. Kalouptsidis and S. Theodoridis, *Adaptive Systems Identification and Signal Processing Algorithms*, Englewood Cliffs, NJ: Prentice Hall, 1993.

[2] Maurice G. Bellanger, "The FLS-QR algorithm for adaptive filtering: The case of multichannel signals," *Signal Processing*, vol. 22, pp. 115–126, 1991.

[3] A. A. Rontogiannis and S. Theodoridis, "Multichannel fast QRD-LS adaptive filtering: New technique and algorithms," *IEEE Transactions on Signal Processing*, vol. 46, pp. 2862–2876, November 1998.

[4] J. A. Apolinário, M. G. Siqueira, and P. S. R. Diniz, "Fast QR Algorithms Based on Backward Prediction Errors: A New Implementation and Its Finite Precision Performance," *Birkhäuser, Systems, and Signal Processing*, vol. 22, no. 4, pp. 335–349, July/August 2003.

[5] C. A. Medina S., J. A. Apolinário Jr., and M. G. Siqueira, "A unified framework for multichannel fast QRD-LS adaptive filters based on backward prediction errors," *MWSCAS'O2, Tulsa–USA*, vol. 3, August 2002.

[6] A. L. L. Ramos and J. A. Apolinário Jr., "A lattice version of the multichannel FQRD algorithm based on *a posteriori* backward errors," *11th Internacional Conference on Telecommunications, Fortaleza, Brazil, ICT'2004, LNCS*, vol. 3124, pp. 488–497, August 2004.

[7] A. L. L. Ramos and J. A. Apolinário Jr., "A new multiple order multichannel fast QRD algorithm and its application to non-linear system identification," *XXI Simpósio Brasileiro de Telecomunicações, SBT 2004, Belém, Brazil*, September 2004.

[8] V. John Mathews and Giovanni L. Sicuranza, *Polynomial Signal Processing*, Wiley–Intercience: John Wiley and Sons, 2000.

[9] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementations*, 2nd Edition, Kluwer Academic Publishers, Boston, 2002.

[10] S. T. Alexander and A. L. Ghirnikar, "A method for recursive least squares filtering based upon an inverse QR decomposition," *IEEE Transactions on Signal Processing*, vol. 41, pp. 20–30, January 1993.