# A UNIFIED FRAMEWORK FOR MULTICHANNEL FAST QRD-LS ADAPTIVE FILTERS BASED ON BACKWARD PREDICTION ERRORS

*César A. Medina S.*, José A. Apolinário Jr.†, and Marcio G. Siqueira*

IME – Department of Electrical Engineering
Praça General Tibúrcio, 80 – Urca
Rio de Janeiro, RJ 22290-270 Brazil
csmedina@epq.ime.eb.br,
apolin@ieee.org and mgs@ieee.org

**ABSTRACT**

Fast QR decomposition algorithms based on backward prediction errors are well known for their good numerical behavior and their low complexity when compared to similar algorithms with forward error update. Their application to multiple channel input signals generates more complex equations although the basic matrix expressions are similar. This paper presents a unified framework for a family of multichannel fast QRD-LS algorithms. This family comprises four algorithms — two basic algorithms with two different versions each. These algorithms are detailed in this work.

## 1. INTRODUCTION

Digital processing of multichannel signals using adaptive filters has recently found a variety of new applications [1] such as 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. This increased number of applications has spawned a renewed interest in efficient multichannel algorithms. One class of algorithms, known as multichannel fast QR decomposition least-squares adaptive algorithms based on backward error updating, has become an attractive option because of fast convergence properties and reduced computational complexity.

In the case of one single channel, a unified formulation for Fast QRD-LS algorithms is available in [2]. In this paper, the basic algorithm proposed in [3] is studied and a new formulation is developed for the family of multichannel fast QRD algorithm based on **backward** errors.

This paper is organized as follows. Is Section 2 we present the common matrix equations for the two Multichannel Fast QRD-LS algorithms. Section 3 presents the named MC FQR_PRI_B based em *a priori* backward errors updating. Section 4 addresses the MC FQR_POS_B (*a posteriori* backward errors updating). In Section 5 we present two different versions for each algorithm and, among them, two algorithms that hadn't been proposed before. Section 6 shows computer simulations with one of the proposed algorithms. Conclusions are summarized in Section 7.

*This work was partially supported by CAPES, Brazil.

†Supported by the FAPERJ from Brazil (grant no. E-26/171.307/2001)

## 2. THE MULTICHANNEL ALGORITHM: COMMON EQUATIONS

In this work $N$ is defined as the number of filter coefficients and $M$ is the number of input channels. The objective function to be minimized according to the least-squares (LS) algorithm is defined as

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

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

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

and

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

where $\boldsymbol{x}_k^T = [x_1(k) \quad x_2(k) \quad \cdots \quad x_M(k)]$ is the input vector at time instant $k$.

If we have $\boldsymbol{U}_N(k)$ as the Cholesky factor of $\boldsymbol{X}_N(k)$, obtained through Givens rotation matrix $\boldsymbol{Q}_N(k)$, then

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

Assuming the use of forward prediction, we define the matrix $\boldsymbol{X}_{N+1}(k)$ as follows.

$$\boldsymbol{X}_{N+1}(k) = \begin{bmatrix} \boldsymbol{d}_f(k) & \begin{matrix} \boldsymbol{X}_N(k-1) \\ \boldsymbol{0}^T \end{matrix} \\ \multicolumn{2}{c}{\boldsymbol{0}_{(M-1)\mathbf{x}(MN+M)}} \end{bmatrix} \tag{5}$$

where $\boldsymbol{d}_f(k) = [\boldsymbol{x}_k \quad \lambda^{1/2}\boldsymbol{x}_{k-1} \cdots \lambda^{k/2}\boldsymbol{x}_0]^T$ is the $(k+1) \times M$ forward reference signal.

$\boldsymbol{U}_{N+1}(k)$ can be obtained by applying Givens rotations to (5) as follows

$$\begin{aligned} \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U}_{N+1}(k) \end{bmatrix} &= \boldsymbol{Q}'_f(k)\boldsymbol{Q}_f(k) \begin{bmatrix} \boldsymbol{Q}(k-1) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{M\mathbf{x}M} \end{bmatrix} \boldsymbol{X}_{N+1}(k) \\ &= \boldsymbol{Q}'_f(k)\boldsymbol{Q}_f(k) \begin{bmatrix} \boldsymbol{Q}(k-1) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{M\mathbf{x}M} \end{bmatrix} \begin{bmatrix} \boldsymbol{d}_f(k) & \begin{matrix} \boldsymbol{X}_N(k-1) \\ \boldsymbol{0}^T \end{matrix} \\ \multicolumn{2}{c}{\boldsymbol{0}_{(M-1)\mathbf{x}(MN+M)}} \end{bmatrix} \\ &= \boldsymbol{Q}'_f(k)\boldsymbol{Q}_f(k) \begin{bmatrix} \boldsymbol{e}_{fq1}(k) & \boldsymbol{0} \\ \boldsymbol{d}_{fq2} & \boldsymbol{U}_N(k-1) \\ \lambda^{1/2}\boldsymbol{x}_0^T & \boldsymbol{0}^T \\ \multicolumn{2}{c}{\boldsymbol{0}_{(M-1)\mathbf{x}(MN+M)}} \end{bmatrix} \\ &= \boldsymbol{Q}'_f(k) \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{d}_{fq2}(k) & \boldsymbol{U}_N(k-1) \\ \boldsymbol{E}_f(k) & \boldsymbol{0}^T \end{bmatrix} \end{aligned} \tag{6}$$

Where $\boldsymbol{Q}_f(k)$ and $\boldsymbol{Q'}_f(k)$ are series of Givens rotations that zeroes the first $k - MN$ rows and performs a triangularization process of (5).

The resulting *null* section can be removed as shown below:

$$\boldsymbol{U}_{N+1}(k) = \boldsymbol{Q'}_{\theta f}(k) \begin{bmatrix} \boldsymbol{d}_{fq2}(k) & \boldsymbol{U}_N(k-1) \\ \boldsymbol{E}_f(k) & \mathbf{0} \end{bmatrix} \tag{7}$$

where $\boldsymbol{Q'}_{\theta f}(k)$ is the fixed range matrix of $\boldsymbol{Q'}_f(k)$.

Based upon the above equation it is possible to obtain

$$[\boldsymbol{U}_{N+1}(k)]^{-1} = \begin{bmatrix} \mathbf{0} & \boldsymbol{E}_f^{-1}(k) \\ \boldsymbol{U}_N^{-1}(k-1) & -\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{d}_{fq2}(k)\boldsymbol{E}_f^{-1}(k) \end{bmatrix} \boldsymbol{Q'}^T_{\theta f}(k) \tag{8}$$

and define $\boldsymbol{E}_f^0(k)$ as the zero order forward error covariance matrix as,

$$\begin{bmatrix} \mathbf{0} \\ \boldsymbol{E}_f^0(k) \end{bmatrix} = \boldsymbol{Q'}_{\theta f}(k) \begin{bmatrix} \boldsymbol{d}_{fq2}(k) \\ \boldsymbol{E}_f(k) \end{bmatrix} \tag{9}$$

From the fact that

$$\boldsymbol{Q}_f(k) = \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \tag{10}$$

and

$$\boldsymbol{Q}(k-1) = \hat{\boldsymbol{Q}}(k-1) \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}(k-2) \end{bmatrix} \tag{11}$$

and knowing that

$$\begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{MxM} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_{MxM} \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \tag{12}$$

we could write from (6)

$$\begin{aligned} \begin{bmatrix} \mathbf{0} \\ \boldsymbol{d}_{fq2}(k) \\ \boldsymbol{E}_f(k) \end{bmatrix} &= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{I}_{MxM} \end{bmatrix} \begin{bmatrix} \mathbf{1} & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}(k-2) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I}_{MxM} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^T \\ \lambda^{1/2}\boldsymbol{x}_{k-1}^T \\ \vdots \\ \lambda^{k/2}\boldsymbol{x}_0^T \\ \mathbf{0}_{M-1xM} \end{bmatrix} \\ &= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{I}_{MxM} \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{Q}_f(k-1) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^T \\ \lambda^{1/2}\boldsymbol{e}_{fq1}(k-1) \\ \lambda^{1/2}\boldsymbol{d}_{fq2}(k-1)\lambda^{k/2}\boldsymbol{x}_0^T \\ \mathbf{0}_{M-1xM} \end{bmatrix} \\ &= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{I}_{MxM} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k^T \\ \mathbf{0} \\ \lambda^{1/2}\boldsymbol{d}_{fq2}(k-1) \\ \lambda^{1/2}\boldsymbol{E}_f(k-1) \end{bmatrix} \\ &= \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \hat{\boldsymbol{Q}}(k-1) \begin{bmatrix} \boldsymbol{x}_k^T \\ \mathbf{0} \\ \lambda^{1/2}\boldsymbol{d}_{fq2}(k-1) \end{bmatrix} \\ \lambda^{1/2}\boldsymbol{E}_f(k-1) \end{bmatrix} \end{aligned} \tag{13}$$

from (6) and (10) could be show that

$$\begin{bmatrix} \mathbf{0} \\ \boldsymbol{d}_{fq2}(k) \\ \boldsymbol{E}_f(k) \end{bmatrix} = \hat{\boldsymbol{Q}}_f(k) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^T(k) \\ \mathbf{0} \\ \boldsymbol{d}_{fq2}(k) \\ \lambda^{1/2}\boldsymbol{E}_f(k-1) \end{bmatrix} \tag{14}$$

from (13) and (14) and using fixed order matrix $\boldsymbol{Q}_\theta(k)$ from $\hat{\boldsymbol{Q}}(k)$.

$$\begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^T(k+1) \\ \boldsymbol{d}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}_{k+1}^T \\ \lambda^{1/2}\boldsymbol{d}_{fq2}(k) \end{bmatrix} \tag{15}$$

Similarly, from (14) it is possible to obtain

$$\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{E}_f(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^T(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f(k) \end{bmatrix} \tag{16}$$

Finally, the joint process estimation is performed by the following expressions [1]

$$\begin{bmatrix} \boldsymbol{e}_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} d(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k) \end{bmatrix} \tag{17}$$

$$e'(k) = e_{q1}(k)/\gamma(k) = e(k)/\gamma^2(k) \tag{18}$$

## 3. THE MULTICHANNEL FQR_PRI_B ALGORITHM

In the MC FQR_PRI_B algorithm, the *a priori* backward prediction error vector is updated as follows [4].

$$\boldsymbol{a}_N(k) = \lambda^{1/2}\boldsymbol{U}_N^{-T}(k-1)\boldsymbol{x}_N(k) \tag{19}$$

From the definition above and (8) it is possible to obtain

$$\begin{aligned} \boldsymbol{a}_{N+1}(k+1) &= \lambda^{1/2}\boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \mathbf{0} & \boldsymbol{U}_N^{-T}(k-1) \\ \boldsymbol{E}_f^{-T}(k) & -\boldsymbol{E}_f^{-T}(k)\boldsymbol{d}_{fq2}^T(k)\boldsymbol{U}_N^{-T}(k-1) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{x}_N(k) \end{bmatrix} \\ &= \lambda^{1/2}\boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \boldsymbol{U}_N^{-T}(k-1)\boldsymbol{x}_N(k) \\ \boldsymbol{E}_f^{-T}(k)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-T}(k)\boldsymbol{d}_{fq2}^T(k)\boldsymbol{U}_N^{-T}(k-1)\boldsymbol{x}_N(k) \end{bmatrix} \\ &= \boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \boldsymbol{a}_N(k) \\ \boldsymbol{r}(k+1) \end{bmatrix} \end{aligned} \tag{20}$$

where

$$\begin{aligned} \boldsymbol{r}(k+1) &= \lambda^{1/2}\left[\boldsymbol{E}_f^{-T}(k)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-T}(k)\boldsymbol{d}_{fq2}^T(k)\boldsymbol{U}_N^{-T}(k-1)\boldsymbol{x}_N(k)\right] \\ &= \lambda^{1/2}\boldsymbol{E}_f^{-T}(k)\left[\boldsymbol{x}_{k+1} - \left[\boldsymbol{U}_N^{-1}(k-1)\boldsymbol{d}_{fq2}(k)\right]^T\boldsymbol{x}_N(k)\right] \\ &= \lambda^{1/2}\boldsymbol{E}_f^{-T}(k)\left[\boldsymbol{x}_{k+1} - \boldsymbol{W}_f^T\boldsymbol{x}_N(k)\right] \\ &= \lambda^{1/2}\boldsymbol{E}_f^{-T}(k)\left[\boldsymbol{x}_{k+1} - \boldsymbol{d}_f(k)\right] \\ &= \lambda^{1/2}\boldsymbol{E}_f^{-T}(k)\widetilde{\boldsymbol{e}}'_f(k+1) \end{aligned} \tag{21}$$

finally [4]

$$\widetilde{\boldsymbol{e}}'_f(k+1) = \gamma(k)\widetilde{\boldsymbol{e}}_{fq1}(k+1) \tag{22}$$

The following equation is used to update $\boldsymbol{Q}_\theta(k+1)$[4]:

$$\begin{bmatrix} 1/\gamma(k+1) \\ \mathbf{0} \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} 1 \\ -\boldsymbol{a}(k+1) \end{bmatrix} \tag{23}$$

Equation (21) requires a matrix inversion operation which can be numerically unstable. In order to avoid stability problems we can use [3]

$$\begin{bmatrix} * \\ \mathbf{0} \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} 1/\gamma(k+1) \\ -\boldsymbol{r}(k+1) \end{bmatrix} \tag{24}$$

The resulting equations are summarized in Table 1.

Table 1: The Multichannel FQR_PRI_B Equations.

| **MC FQR_PRI_B** |
|---|
| 1. Obtaining $\boldsymbol{d}_{fq2}(k+1)$ |
| $\begin{bmatrix} \widetilde{\boldsymbol{e}}^T_{fq1}(k+1) \\ \boldsymbol{d}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k) \begin{bmatrix} \boldsymbol{x}^T_{k+1} \\ \lambda^{1/2}\boldsymbol{d}_{fq2}(k) \end{bmatrix}$ |
| 2. Obtaining $\boldsymbol{E}_f(k+1)$ |
| $\begin{bmatrix} \mathbf{0}^T \\ \boldsymbol{E}_f(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_f(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}^T_{fq1}(k+1) \\ \lambda^{1/2}\boldsymbol{E}_f(k) \end{bmatrix}$ |
| 3. Obtaining $\boldsymbol{a}_N(k+1)$ |
| $\boldsymbol{a}_{N+1}(k+1) = \boldsymbol{Q}'_{\theta f}(k) \begin{bmatrix} \boldsymbol{a}_N(k) \\ \boldsymbol{r}(k+1) \end{bmatrix}$ |
| 4. Obtaining $\boldsymbol{Q}'_{\theta f}(k+1)$ |
| $\begin{bmatrix} \mathbf{0} \\ \boldsymbol{E}^0_f(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{d}_{fq2}(k+1) \\ \boldsymbol{E}_f(k+1) \end{bmatrix}$ |
| 5. Obtaining $\boldsymbol{Q}_\theta(k+1)$ |
| $\begin{bmatrix} 1/\gamma(k+1) \\ \mathbf{0} \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} 1 \\ -\boldsymbol{a}(k+1) \end{bmatrix}$ |
| 6. Joint Estimation |
| $\begin{bmatrix} \boldsymbol{e}_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}_\theta(k+1) \begin{bmatrix} d(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k) \end{bmatrix}$ |
| 7. Obtaining the *a priori* error |
| $e'(k+1) = e_{q1}(k+1)/\gamma(k+1)$ |

## 4. THE MULTICHANNEL FQR_POS_B ALGORITHM

To derive equations for this algorithm, we update the *a posterior* backward prediction errors vector as in [4]

$$\boldsymbol{f}_{n+1}(k+1) = \boldsymbol{U}^{-T}_{N+1}(k+1)\boldsymbol{X}_{N+1}(k+1) \tag{25}$$

From (8) and the expression above, we obtain

$$\begin{aligned}
\boldsymbol{f}_{N+1}(k+1) &= \boldsymbol{Q'}_{\theta f}(k+1)\begin{bmatrix} \mathbf{0} & \boldsymbol{U}_N^{-T}(k) \\ \boldsymbol{E}_f^{-T}(k+1) & -\boldsymbol{E}_f^{-T}(k+1)\boldsymbol{d}_{fq2}^T(k+1)\boldsymbol{U}_N^{-T}(k) \end{bmatrix}\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{x}_N(k) \end{bmatrix} \\
&= \boldsymbol{Q'}_{\theta f}(k+1)\begin{bmatrix} \boldsymbol{U}_N^{-T}(k)\boldsymbol{x}_N(k) \\ \boldsymbol{E}_f^{-T}(k+1)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-T}(k+1)\boldsymbol{d}_{fq2}^T(k+1)\boldsymbol{U}_N^{-T}(k)\boldsymbol{x}_N(k) \end{bmatrix} \\
&= \boldsymbol{Q}'_{\theta f}(k+1)\begin{bmatrix} \boldsymbol{f}_N(k) \\ \boldsymbol{p}(k+1) \end{bmatrix}
\end{aligned} \tag{26}$$

where

$$\begin{aligned}
\boldsymbol{p}(k+1) &= \left[\boldsymbol{E}_f^{-T}(k+1)\boldsymbol{x}_{k+1} - \boldsymbol{E}_f^{-T}(k+1)\boldsymbol{d}_{fq2}^T(k+1)\boldsymbol{U}_N^{-T}(k)\boldsymbol{x}_N(k)\right] \\
&= \boldsymbol{E}_f^{-T}(k+1)\left[\boldsymbol{x}_{k+1} - \left[\boldsymbol{U}_N^{-1}(k)\boldsymbol{d}_{fq2}(k+1)\right]^T\boldsymbol{x}_N(k)\right] \\
&= \boldsymbol{E}_f^{-T}(k+1)\left[\boldsymbol{x}_{k+1} - \boldsymbol{W}_f^T\boldsymbol{x}_N(k)\right] \\
&= \boldsymbol{E}_f^{-T}(k)\left[\boldsymbol{x}_{k+1} - \boldsymbol{d}_f(k)\right] \\
&= \boldsymbol{E}_f^{-T}(k+1)\widetilde{\boldsymbol{e}}'_f(k+1)
\end{aligned} \tag{27}$$

Also from [4], we have the following expression, similar to its single dimension counterpart.

$$\boldsymbol{Q}_\theta(k+1)\begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \gamma(k+1) \\ \boldsymbol{f}_N(k+1) \end{bmatrix} \tag{28}$$

The complete algorithm is summarized in Table 2.

## 5. THE DIFFERENT VERSIONS

The different versions presented in this section are based on the implementation of a particular matrix equation. For both algorithms we have a matrix equation with the following structure (step $3$ and $4$ in Tables $1$ and $2$, respectively)

$$\boldsymbol{u}(k+1) = \boldsymbol{Q}'_{\theta f}(k+1)\begin{bmatrix} \boldsymbol{v}(k) \\ \boldsymbol{\beta}(k+1) \end{bmatrix} \tag{29}$$

where $\boldsymbol{\beta}(k+1)$ is a vector of order $M$, and is updated without using any prior knowledge of any element of $\boldsymbol{u}(k+1)$.

For the first version of each algorithm, we assume that the inverse of $\boldsymbol{Q'}_{\theta f}(k+1)$ corresponds to $\boldsymbol{Q'}^T_{\theta f}(k+1)$.

Table 2: The Multichannel FQR_POS_B Equations.

| MC FQR_POS_B |
|---|
| 1. Obtaining $\boldsymbol{d}_{fq2}(k+1)$ |
| $\begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^{T}(k+1) \\ \boldsymbol{d}_{fq2}(k+1) \end{bmatrix} = \boldsymbol{Q}_{\theta}(k) \begin{bmatrix} \boldsymbol{x}_{k+1}^{T} \\ \lambda^{1/2}\boldsymbol{d}_{fq2}(k) \end{bmatrix}$ |
| 2. Obtaining $\boldsymbol{E}_{f}(k+1)$ |
| $\begin{bmatrix} \boldsymbol{0}^{T} \\ \boldsymbol{E}_{f}(k+1) \end{bmatrix} = \overline{\boldsymbol{Q}}_{f}(k+1) \begin{bmatrix} \widetilde{\boldsymbol{e}}_{fq1}^{T}(k+1) \\ \lambda^{1/2}\boldsymbol{E}_{f}(k) \end{bmatrix}$ |
| 3. Obtaining $\boldsymbol{Q}'_{\theta f}(k+1)$ |
| $\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{E}_{f}^{0}(k+1) \end{bmatrix} = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{d}_{fq2}(k+1) \\ \boldsymbol{E}_{f}(k+1) \end{bmatrix}$ |
| 4. Obtaining $\boldsymbol{f}_{N}(k+1)$ |
| $\boldsymbol{f}_{N+1}(k+1) = \boldsymbol{Q}'_{\theta f}(k+1) \begin{bmatrix} \boldsymbol{f}_{N}(k) \\ \boldsymbol{p}(k+1) \end{bmatrix}$ |
| 5. Obtaining $\boldsymbol{Q}_{\theta}(k+1)$ |
| $\boldsymbol{Q}_{\theta}(k+1) \begin{bmatrix} 1 \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \gamma(k+1) \\ \boldsymbol{f}_{N}(k+1) \end{bmatrix}$ |
| 6. Joint Estimation |
| $\begin{bmatrix} \boldsymbol{e}_{q1}(k+1) \\ \boldsymbol{d}_{q2}(k+1) \end{bmatrix} = \boldsymbol{Q}_{\theta}(k+1) \begin{bmatrix} d(k+1) \\ \lambda^{1/2}\boldsymbol{d}_{q2}(k) \end{bmatrix}$ |
| 7. Obtaining the *a priori* error |
| $e'(k+1) = e_{q1}(k+1)/\gamma(k+1)$ |

Therefore, it is possible to show that

$$\begin{bmatrix} \boldsymbol{v}_1(k+1) \\ \vdots \\ \boldsymbol{v}_N(k+1) \\ \boldsymbol{\beta}(k+1) \end{bmatrix} = \prod_{L1}\prod_{L2} \begin{bmatrix} \boldsymbol{I}_j & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & cos'_{\theta Mj+s+1} & \boldsymbol{0} & sin'_{\theta Mj+s+1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_{MN+M-j-s-2} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & -sin'_{\theta Mj+s+1} & \boldsymbol{0} & cos'_{\theta Mj+s+1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I}_s \end{bmatrix} \times \begin{bmatrix} \boldsymbol{u}_0(k+1) \\ \boldsymbol{u}_1(k+1) \\ \vdots \\ \boldsymbol{u}_N(k+1) \end{bmatrix} \tag{30}$$

where $L1$ is from $j = 0$ to $MN - 1$ and $L2$ is from $s = 0$ to $M - 1$. It is possible to update $\boldsymbol{v}(k)$ based upon $\boldsymbol{u}(k+1)$ if $\boldsymbol{u}_{N+1}(k+1)$ is previously known.

For the second version of this algorithm, equation (29) is used directly. The resulting operations are shown below

$$\begin{bmatrix} \boldsymbol{u}_0(k+1) \\ \boldsymbol{u}_1(k+1) \\ \vdots \\ \boldsymbol{u}_N(k+1) \end{bmatrix} =$$

$$\prod_{L3}\prod_{L4} \begin{bmatrix} \boldsymbol{I}_j & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & cos'_{\theta Mj+s+1} & \mathbf{0} & -sin'_{\theta Mj+s+1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I}_{MN+M-j-s-2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & sin'_{\theta Mj+s+1} & \mathbf{0} & cos'_{\theta Mj+s+1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{I}_s \end{bmatrix}$$

$$\times \begin{bmatrix} \boldsymbol{v}_1(k+1) \\ \vdots \\ \boldsymbol{v}_N(k+1) \\ \boldsymbol{\beta}(k+1) \end{bmatrix} \tag{31}$$

where $L3$ is from $j = MN - 1$ to $0$ and $L4$ is from $s = M - 1$ to $0$.

## 5.1. The Multichannel FQR_PRI_B: Versions 1 and 2

Another way to derive (20) is as follows. From (7) we can get

$$\boldsymbol{U}_{N+1}(k) = \begin{bmatrix} \mathbf{0} & \boldsymbol{B} \\ \boldsymbol{E}_f^0(k) & \boldsymbol{C} \end{bmatrix} \tag{32}$$

Obtaining the inverse

$$[\boldsymbol{U}_{N+1}(k)]^{-1} = \begin{bmatrix} -\left[\boldsymbol{E}_f^0(k)\right]^{-1}\boldsymbol{C}\boldsymbol{B}^{-1} & \left[\boldsymbol{E}_f^0(k)\right]^{-1} \\ \boldsymbol{B}^{-1} & \mathbf{0} \end{bmatrix} \tag{33}$$

From (19) and using (33)

$$\begin{aligned} \boldsymbol{a}_{N+1}(k+1) &= \lambda^{1/2} \begin{bmatrix} \left[-\left[\boldsymbol{E}_f^0(k)\right]^{-1}\boldsymbol{C}\boldsymbol{B}^{-1}\right]^T & \boldsymbol{B}^{-T} \\ \left[\boldsymbol{E}_f^0(k)\right]^{-T} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{x}_N(k) \end{bmatrix} \\ &= \begin{bmatrix} * \\ \lambda^{1/2}\left[\boldsymbol{E}_f^0(k)\right]^{-T}\boldsymbol{x}_{k+1} \end{bmatrix} \end{aligned} \tag{34}$$

It could be see that the last *element* of $\boldsymbol{a}_N(k+1)$ was known before its updating process.

This property is used in MC FQR_PRI_B_1, which is the first algorithm version introduced in this paper. Table 3 shows pseudo code for MC FQR_PRI_B step 3 using Matlab© (*Obtaining* $\boldsymbol{a}_N(k+1)$).

The second version of the MC FQR_PRI_B algorithm is based on the direct computation of $\boldsymbol{a}_N(k+1)$ according to the matrix equation. This version was introduced in [3] and the corresponding implementation is presented in Table 4.

## 5.2. The Multichannel FQR_POS_B: Versions 1 and 2

Deriving (26) in the same way as (34), it could be show that the last *element* of $\boldsymbol{f}_N(k+1)$ was known prior to its updating process and was equal to $\left[\boldsymbol{E}_f^0(k+1)\right]^{-T}\boldsymbol{x}_{k+1}$. This property generates the first algorithm version

Table 3: The MC FQR_PRI_B: Version 1.

```
⋮
aux = inv(Ef_0)' * x(k+1,:)'/(lambda ^ (.5));
tempa = a;
a(M*(N-1)+1:L) = aux;
for j = 1:N-1
  for i = 1:M
   for s = 1:M
    a(M*(N-j)-i+1)=(tempa(M*(N-j+1)-i+1)-aux(M-s+1)*sin_p(M*(N-j+1)-i+1,s))...
     /cos_p(M*(N-j+1)-i+1,s);
    tempa(M*(N-j+1)-i+1)=a(M*(N-j)-i+1);
    aux(M-s+1)=-a(M*(N-j)-i+1)*sin_p(M*(N-j+1)-i+1,s)+aux(M-s+1)...
     *cos_p(M*(N-j+1)-i+1,s);
   end
  end
end
⋮
```

Table 4: The MC FQR_PRI_B: Version 2.

```
⋮
for j = 1:N
  for i = 1:M
   for s = 1:M
    temp1 = a (M*j + i);
    temp2 = r(M - s + 1);
    a (M*(j-1)+i)=temp1*cos_p(M*(j-1)+i,s)-temp2*sin_p(M*(j-1)+i,s);
    a (M*j + i)=a (M*(j-1)+i);
    r (M-s+1)=temp1*sin_p(M*(j-1)+i,s)+temp2*cos_p(M*(j-1)+i,s);
   end
  end
end
a(M*N + 1:M*N + M) = r;
⋮
```

known as the MC FQR_POS_B_1. Table 5 presents the implementation of its step 4 (*Obtaining* $\boldsymbol{f}_N(k+1)$). The second version of the Multichannel FQR_POS_B algorithm is based on the direct computation of $\boldsymbol{f}_N(k+1)$ according to the matrix equation. This version was introduced in [5] and its implementation is presented in Table 6.

## 6. SIMULATIONS

This section presents simulation results of the Multichannel FQR_POS_B algorithm version 1. The algorithm was used in a system identification problem with $M = 5$ channels, each one with $N = 10$ coefficients. The input signal to each channel was colored noise with eigenvalue spread equal to $350$ and observed noise with variance corresponding to $-40dB$. The forgetting factor was set to $\lambda = 0.95$. Fig. 1 presents the MSE (in $dB$) over an ensemble of $20$ independent runs. It is worth noting that all other versions present identical learning curves in infinite precision.

Table 5: The MC FQR_POS_B: Version 1.

```
⋮
aux = inv(Ef_0)' * x(k+1,:)';
tempf = f;
f(M*(N-1)+1:L) = aux;
for j = 1:N-1
  for i = 1:M
   for s = 1:M
    f(M*(N-j)-i+1)=(tempf(M*(N-j+1)-i+1)-aux(M-s+1)*sin_p(M*(N-j+1)-i+1,s))...
     /cos_p(M*(N-j+1)-i+1,s);
    tempf(M*(N-j+1)-i+1) = f(M*(N-j)-i+1);
    aux(M-s+1) = -f(M*(N-j)-i+1)*sin_p(M*(N-j+1)-i+1,s)+aux(M-s+1)...
     *cos_p(M*(N-j+1)-i+1,s);
   end
  end
end
⋮
```

Table 6: The MC FQR_POS_B: Version 2.

```
⋮
p = inv(Ef)'*efq1' * gamma;
for j = 1:N
  for i = 1:M
   for s = 1:M
    temp1 = f (M * j + i);
    temp2 = p(M - s + 1);
    f(M*(j - 1)+i)=temp1*cos_p(M*(j-1)+i,s)-temp2*sin_p(M*(j-1)+i,s);
    f(M*j + i) = f (M*(j - 1) + i);
    p(M-s+1)=temp1*sin_p(M*(j-1)+i,s)+temp2*cos_p(M*(j-1)+i,s);
   end
  end
end
f(M * N + 1:M*N + M) = p;
⋮
```

## 7. CONCLUSIONS

This paper presents a unified framework for Multichannel FQR algorithms based on backward error updating. Different algorithms were derived using the same notation, therefore highlighting their similarities and differences. [1] Alternative and more efficient forms — such as the MC FQR_PRI_B_2 algorithm in [3] — are currently under investigation.

[1]Matlab© codes, are available for downloading via internet at the web page http://www.ime.eb.br/~apolin

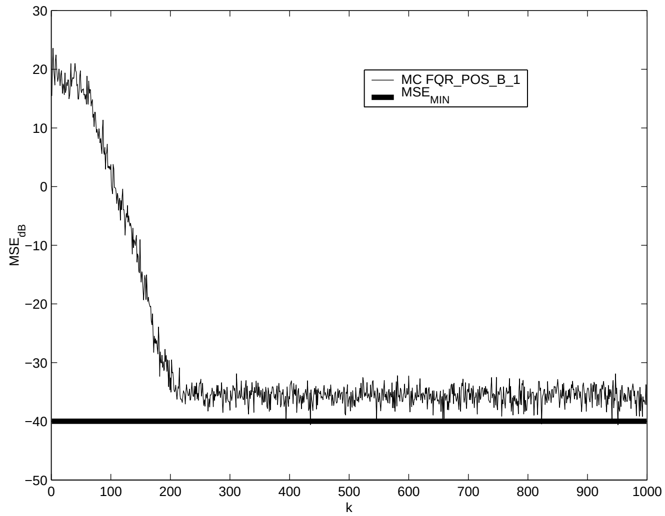


Figure 1: Learning Curve of the MC FQR_POS_B, Version 1.

## 8. REFERENCES


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

[2] J. A. Apolinário Jr., M. G. Siqueira, and P. S. R. Diniz, "On Fast QR Algorithms Based on Backward Prediction errors: New Results and Comparisons," in *Proc. First Balkan Conf. on Signal Process., Commun., Circuits, and Systems*, Istanbul, Turkey, June 2000.

[3] A. A. Rontogiannis and S. Theodoridis, "Multichannel Fast QRD-LS Adaptive Filtering: New Technique and Algorithms," *IEEE Transactions on Signal Processing*, vol. 46, No. 11, Nov. 1998.

[4] J. A. Apolinário Jr., *New Algorithms of Adaptive Filtering: LMS with Data-Reusing and Fast RLS Based on QR Decomposition*, D.Sc. Thesis, COPPE/UFRJ, Rio de Janeiro, RJ, Brazil, 1998.

[5] M. G. Bellanger and P. A. Regalia, "The FLS-QR algorithm for adaptive filtering: The case of multichannel signal," Signal Process. vol. 22, pp. 115–126, 1991.