RLS Adaptive Filtering Algorithms Based on Parallel Computations

Victor I. DJIGAN

VLSI Development Laboratory, ELVEES R&D Center of Microelectronics, POB 19, Centralny Prospect, Zelenograd, Moscow, Russia 124460

djigan@elvees.ru

Abstract. *The paper presents a family of the sliding window RLS adaptive filtering algorithms with the regularization of adaptive filter correlation matrix. The algorithms are developed in forms, fitted to the implementation by means of parallel computations. The family includes RLS and fast RLS algorithms based on generalized matrix inversion lemma, fast RLS algorithms based on square root free inverse QR decomposition and linearly constrained RLS algorithms. The considered algorithms are mathematically identical to the appropriate algorithms with sequential computations. The computation procedures of the developed algorithms are presented. The results of the algorithm simulation are presented as well.*

Keywords

Adaptive filtering, RLS, fast RLS, QR decomposition, linear constraints, parallel computations.

1. Introduction

Adaptive signal processing [1] is an essential part of modern digital signal processing. Theoretical results, obtained in the field, are widely used in adaptive filters design. Communication channels equalization, echo cancellation, suppression of spatially separated noise sources – these are only a few examples of the practical use of such filters [2]. The efficiency of the applications depends on the algorithms that the adaptive filters are based on.

In the applications the simplest gradient adaptive filtering algorithms are basically used because the algorithms have small arithmetic complexity. However, the filters, which use such algorithms, are not good enough for the processing of non-stationary signals. The Recursive Least Squares (RLS) algorithms [3] are more appropriate for the processing of the signals, but they require more computing resources. The algorithms complexity is not a problem in modern Digital Signal Processors (DSP) anymore, as the devices have enough resources for the implementation of such algorithms [4]. Besides, as a few DSPs can be integrated in a chip, such chips can be used for compact implementation of signal processing algorithms based on parallel computations. Due to the opportunity, the development of parallel adaptive filtering algorithms becomes an important task.

2. Problem Formulation and Solution

The given paper presents a method of RLS algorithms description, which allows the development of the algorithms in forms, fitted to the implementation by means of parallel computations. The Sliding Window (SW) RLS algorithms with the regularization of correlation matrix for multichannel adaptive filters with unequal number of complex-valued weights in channels are considered. The algorithms diversity includes unconstrained and constrained RLS algorithms, based on generalized Matrix Inversion Lemma (MIL) and square root free inverse QR decomposition (QRD).

Most of RLS algorithms are based on the use of MIL for the recursive inversion of adaptive filter correlation matrix. To provide the tracking properties of the filters, when non-stationary signals are processed, exponential weighting of the signals or (and) SW is used. In SW case, MIL is applied twice per sample. Due to the limited number of samples involved in the estimation of correlation matrix, SW RLS algorithms can be unstable sometimes. Dynamic regularization of the matrix can be used to stabilize RLS algorithms [5]. The SW and regularization are the reasons of the increased arithmetic complexity in comparison with growing window (Prewindowed, PW) RLS algorithms or the absence of the regularization.

The computation load of the complex adaptive filtering algorithms implementation can be decreased by means of parallel computations. To get the parallel RLS algorithms, methods [6] can be used. Based on the methods, parallel algorithms fitted to the implementation by means of two or four processors were developed [6-8].

This paper considers another simple method of the RLS algorithms description, which allows the developing of the regularized PW RLS, SW RLS and regularized SW RLS algorithms of adaptive filtering, including Linearly Constrained (LC) versions of the algorithms, [9-12] as the sequence of the same parallel computations, similar to those of the same named PW RLS algorithms.

A block-diagram of M-channel adaptive filter is shown in Fig. 1. The filter can have unequal number of weights in channels.

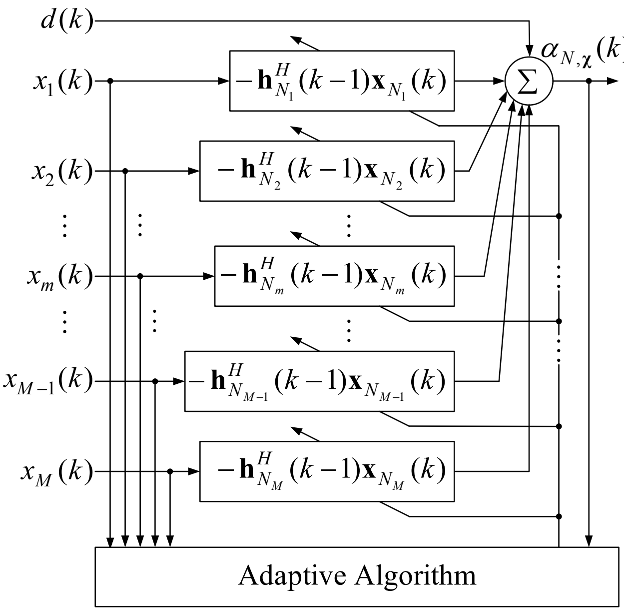

Fig. 1. Multichannel adaptive filter.

The objective of the LC SW least square filtering is to minimize the energy of the error between the desired signal $d(k)$ and the adaptive filter output:

$$E_N(k) = \sum_{i=k-L+1}^{k} \lambda^{k-i}\left[d(i) - \mathbf{h}_N^H(k)\boldsymbol{\chi}_N(i)\right]^2, \qquad (1)$$

where the error is measured over an observation window L samples long. The minimization is carried out under the condition $\mathbf{C}_{NJ}^H \mathbf{h}_{N,\chi}(k) = \mathbf{f}_J$. Here, $\mathbf{h}_N^H(k) = \left[\mathbf{h}_{N_1}^H(k), \mathbf{h}_{N_2}^H(k), \ldots, \mathbf{h}_{N_m}^H(k), \ldots, \mathbf{h}_{N_M}^H(k)\right]$ is a vector of M-channel adaptive filter weights; $\mathbf{h}_{N_m}(k) = \left[h_{0,m}, h_{1,m}, \ldots, h_{N_m-1,m}\right]^T$ is a vector of weights in m-th channel of the filter; $\boldsymbol{\chi}_N^T(k) = \left[\mathbf{x}_{N_1}^T(k), \mathbf{x}_{N_2}^T(k), \ldots, \mathbf{x}_{N_m}^T(k), \ldots, \mathbf{x}_{N_M}^T(k)\right]$ is a vector of input signals in the adaptive filter; $\mathbf{x}_{N_m}^T(k) = \left[x_m(k), x_m(k-1), \ldots, x_m(k-N_m+1)\right]$ is a vector of signals in m-th channel; $\mathbf{C}_{NJ}$ and $\mathbf{f}_J$ are matrix and vector of J linear constraints; N_m is a number of weights in m-th channel; $N = \sum_{m=1}^{M} N_m$ is a total number of adaptive filter weights; k is a sample number and λ is a forgetting factor. Superscripts H and T denote Hermitian transpose and transposition of a vector or a matrix; one subscript N, J, or F denotes the dimension of vectors and square matrices, two subscripts NJ or NF denote the dimension of rectangular (non-transposed) matrices.

The solution of the problem (1) is the vector of adaptive filter weights [13]:

$$\mathbf{h}_N(k) = \mathbf{R}_N^{-1}(k)\mathbf{r}_N(k) + \mathbf{R}_N^{-1}(k)\mathbf{C}_{NJ} \times \\ \times \left[\mathbf{C}_{NJ}^H \mathbf{R}_N^{-1}(k)\mathbf{C}_{NJ}\right]^{-1}\left[\mathbf{f}_J - \mathbf{C}_{NJ}^H \mathbf{R}_N^{-1}(k)\mathbf{r}_N(k)\right] \qquad (2)$$

If SW and dynamic regularization are used, the adaptive filter correlation matrix is defined as

$$\mathbf{R}_N(k) = \sum_{i=k-L+1}^{k} \lambda^{k-i}\left[\boldsymbol{\chi}_N(i)\boldsymbol{\chi}_N^H(i) + \xi^2 \boldsymbol{\rho}_N(i) \times \\ \times \boldsymbol{\rho}_N^T(i)\right] = \lambda \mathbf{R}_N(k-1) + \boldsymbol{\chi}_N(k)\boldsymbol{\chi}_N^H(k) - \\ - \mu\boldsymbol{\chi}_N(k-L)\boldsymbol{\chi}_N^H(k-L) + \xi^2 \boldsymbol{\rho}_N(k)\boldsymbol{\rho}_N^T(k) \\ - \mu\xi^2 \boldsymbol{\rho}_N(k-L)\boldsymbol{\rho}_N^T(k-L). \qquad (3)$$

The cross correlation of $\chi_N(k)$ and $d(k)$ is defined as

$$\mathbf{r}_N(k) = \sum_{i=k-L+1}^{k} \lambda^{k-i}\boldsymbol{\chi}_N(i)d^*(i) = \lambda \mathbf{r}_N(k-1) + \\ + \boldsymbol{\chi}_N(k)d^*(k) - \mu\boldsymbol{\chi}_N(k-L)d^*(k-L). \qquad (4)$$

In (3) and (4), $()^*$ means complex conjugate, $\mu=\lambda^L$ and ξ^2 is a small value of a dynamic regularization parameter [5]. Parameter ξ^2 and parameter δ^2 for the initial regularization of correlation matrix are selected as ξ^2, $\delta^2 \geq 0.01\ \sigma^2$, where σ^2 is the variance of adaptive filter input signals. The dynamic regularization vector $\boldsymbol{\rho}_N(k)$ is defined as

$$\boldsymbol{\rho}_N^T(k) = \left[\mathbf{p}_{N_1}^T(k), \mathbf{p}_{N_2}^T(k), \ldots, \mathbf{p}_{N_m}^T(k), \ldots, \mathbf{p}_{N_M}^T(k)\right] \qquad (5)$$

where $\mathbf{p}_{N_m}^T(k) = \left[p_m(k), p_m(k-1), \ldots, p_m(k-N_m+1)\right]$. In (5), $p_m(k)=0$, if $1+n_{\mathrm{mod}\,N_m} \neq 1$, and $p_m(k)=1$, if $1+n_{\mathrm{mod}\,N_m} = 1$.

The first item in equation (2) is the solution of the problem (1) without constraints. The second item is determined by linear constraints. As a result, the RLS algorithms, which compute weight vector (2), also consist of two computational procedures: unconstrained and LC ones.

MIL [1], which is used in sequential RLS algorithms, is expressed as $\mathbf{R}^{-1}=\mathbf{B}^{-1}-\mathbf{B}^{-1}\mathbf{c}A^{-1}\mathbf{d}\mathbf{B}^{-1}$, where $A=\mathbf{d}\mathbf{B}^{-1}\mathbf{c}+1$ and $\mathbf{R}=\mathbf{B}+\mathbf{c}\mathbf{d}$. Here $\mathbf{c}$ and $\mathbf{d}$ are vectors. It allows the using of the lemma sequentially to invert matrix (3).

The application of the MIL [6] allows the getting of parallel forms for regularized and SW RLS algorithms. The MIL [6] is very cumbersome and the resulting mathematical descriptions of parallel algorithms [6-8] are cumbersome as well.

In general case, see [14], MIL is expressed as

$$\mathbf{R}^{-1} = \mathbf{B}^{-1} - \mathbf{B}^{-1}\mathbf{C}\mathbf{A}^{-1}\mathbf{D}\mathbf{B}^{-1} \qquad (6)$$

where $\mathbf{A}=\mathbf{D}\mathbf{B}^{-1}\mathbf{C}+\mathbf{S}$, $\mathbf{C}$ and $\mathbf{D}$ are matrices. The equation (6) is a key tool for the development of the parallel SW regularized RLS algorithms, considered in this paper. To use the equation (6), the matrices $\mathbf{C}=[\mathbf{y},\mathbf{x},\mathbf{y},\mathbf{v}]=[\mu^{0.5}\chi_N(k-L),\chi_N(k),\mu^{0.5}\xi\boldsymbol{\rho}_N(k-L),\xi\boldsymbol{\rho}_N(k)]$, $\mathbf{D}=\mathbf{C}^H$ and $\mathbf{S}=diag(-1,1,-1,1)$

have to be created. The columns in matrix $\mathbf{X}_{NF}(k)$ cause the diversity of RLS algorithms: PW, regularized PW, SW and regularized SW.

3. Parallel RLS Algorithms

The using of the equation (6) allows the getting of a RLS algorithm of adaptive filtering in parallel form:

0) $\textbf{Init.:}\, \boldsymbol{\chi}_N(0)=\mathbf{0}_N,...,\boldsymbol{\chi}_N(0-L+1)=\mathbf{0}_N,$
$\boldsymbol{\rho}_N(0)=\mathbf{0}_N,...,\boldsymbol{\rho}_N(0-L+1)=\mathbf{0}_N,$
$d(0)=0,...,d(0-L+1)=0, \mathbf{X}_{NF}(0)=\mathbf{O}_{NF},$
$\mathbf{R}_N^{-1}(0)=\delta^{-2}\boldsymbol{\Lambda}_N, \mathbf{h}_N(0)=\mathbf{0}_N$
$\boldsymbol{\Lambda}_N=diag(1,\lambda,\ldots,\lambda^{N_1-1},\ldots,1,\lambda,\ldots,\lambda^{N_M-1})$

For $k=1,2,\ldots,K$

1) $\mathbf{G}_{NF}(k)=\dfrac{\mathbf{R}_N^{-1}(k-1)\mathbf{X}_{NF}(k)}{\lambda\mathbf{S}_F+\mathbf{X}_{NF}^H(k)\mathbf{R}_N^{-1}(k-1)\mathbf{X}_{NF}(k)}$

2) $\mathbf{R}_N^{-1}(k)=\lambda^{-1}\left[\mathbf{R}_N^{-1}(k-1)-\mathbf{G}_{NF}(k)\times \times\mathbf{X}_{NF}^H(k)\mathbf{R}_N^{-1}(k-1)\right]$

3) $\boldsymbol{\alpha}_F(k)=\mathbf{d}_F(k)-\mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$

4) $\mathbf{h}_N(k)=\mathbf{h}_N(k-1)+\mathbf{G}_{NF}(k)\boldsymbol{\alpha}_F^H(k)$

End for k

The procedure uses a number of matrix and vector computations that are consequently the vector and scalar ones in the proper sequential RLS algorithms. The matrix of Kalman gains $\mathbf{G}_{NF}(k)$ contains F columns. The columns are computed independently each other. So, the matrix can be computed by means of F processors, i.e. in parallel. In the algorithm the vector $\mathbf{d}_F(k)$ is defined as $[\mu^{0.5}d(k-L),d(k),0,0]$, and error signal at the output of adaptive filter, see Fig. 1, is defined as $\alpha_{N,\chi}(k)=d(k)-\mathbf{h}^H_N(k-1)\chi_N(k)=\alpha_F^{(2)}(k)$, where $\alpha_F^{(2)}(k)$ means the second element of the error vector $\boldsymbol{\alpha}_F(k)$. Vectors $\mathbf{d}_F(k)$ and $\boldsymbol{\alpha}_F(k)$ are row-vectors.

There is a distinction of the algorithm from the sequential RLS algorithms. Denominator in the equation (2) is a scalar variable in a sequential algorithm, while it is a matrix with FxF elements in the parallel algorithm. The matrix ensures the mathematic identity of the sequential and parallel RLS algorithms. Because $F\leq4$, the matrix inversion does not effect the algorithm complexity if $N>>F$. So, the complexity of the parallel RLS algorithm is $O(N^2F)$ arithmetic operations per iteration. Similarly, fast (computationally efficient, $O(NF)$ complexity) parallel RLS algorithms can be developed on basis of use of the least squares linear prediction theory [15]. A parallel version of the multichannel Fast Kalman (FK) algorithm is shown below.

0) $\textbf{Init.:}\, \boldsymbol{\chi}_N(0)=\mathbf{0}_N,...,\boldsymbol{\chi}_N(0-L+1)=\mathbf{0}_N,$
$\boldsymbol{\rho}_N(0)=\mathbf{0}_N,...,\boldsymbol{\rho}_N(0-L+1)=\mathbf{0}_N, d(0)=0,$
$...,d(0-L+1)=0, \mathbf{X}_{NF}(0)=\mathbf{O}_{NF},$
$\mathbf{h}_N(0)=\mathbf{0}_N, E_N^{f(m)}(0)=\delta^2,$
$\mathbf{h}_N^{f(m)}(0)=\mathbf{0}_N, \mathbf{h}_N^{b(m)}(0)=\mathbf{0}_N,$
$m=1,2,\ldots,M, \mathbf{G}_{NF}^{(M)}(1)=\mathbf{O}_{NF}$

For $k=1,2,\ldots,K$

For $m=M,M-1,\ldots,1$

1) $\boldsymbol{\alpha}_F^{f(m)}(k)=\mathbf{x}_F^{(m)}(k)-\mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$

2) $\boldsymbol{\alpha}_F^{b(m)}(k)=\mathbf{x}_F^{(m)}(k-N_m)-\mathbf{h}_N^{b(m)H}(k-1)\times \times\mathbf{X}_{NF}^{(m-1)}(k)$

3) $\mathbf{h}_N^{f(m)}(k)=\mathbf{h}_N^{f(m)}(k-1)+\mathbf{G}_{NF}^{(m)}(k)\boldsymbol{\alpha}_F^{f(m)H}(k)$

4) $\mathbf{e}_F^{f(m)}(k)=\left[\mathbf{x}_F^{(m)}(k)-\mathbf{h}_N^{f(m)H}(k)\mathbf{X}_{NF}^{(m)}(k)\right]\mathbf{S}_F$

5) $E_N^{f(m)}(k)=\lambda E_N^{f(m)}(k-1)+\mathbf{e}_F^{f(m)}(k)\times \times\boldsymbol{\alpha}_F^{f(m)H}(k)$

6) $\mathbf{G}_{(N+1)F}^{(m)}(k)=\left[1, \quad -\mathbf{h}_N^{f(m)H}(k)\right]^H\times \times\mathbf{e}_F^{f(m)}(k)/E_N^{f(m)}(k)+\left[\mathbf{0}_F^T, \quad \mathbf{G}_{NF}^{(m)H}(k)\right]^H$

7) $\mathbf{S}_{N+1}^{(m)}\mathbf{T}_{N+1}^{(m)T}\left\{\mathbf{G}_{(N+1)F}^{(m)}(k)\right\}=\begin{bmatrix}\mathbf{Q}_{NF}^{(m)}(k)\\ \mathbf{q}_F^{(m)}(k)\end{bmatrix}$

8) $\mathbf{G}_{NF}^{(m-1)}(k)=\left[\widetilde{\mathbf{Q}}_{NF}^{(m)}(k)+\mathbf{h}_N^{b(m)}(k-1)\mathbf{q}_F^{(m)}(k)\right]\times \times\left[\mathbf{I}_F-\boldsymbol{\alpha}_F^{b(m)H}(k)\mathbf{q}_F^{(m)}(k)\right]^{-1}$

9) $\mathbf{h}_N^{b(m)}(k)=\mathbf{h}_N^{b(m)}(k-1)+\mathbf{G}_{NF}^{(m-1)}(k)\boldsymbol{\alpha}_F^{b(m)H}(k)$

End for m

10) $\boldsymbol{\alpha}_F(k)=\mathbf{d}_F(k)-\mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$

11) $\mathbf{h}_N(k)=\mathbf{h}_N(k-1)+\mathbf{G}_{NF}^{(0)}(k)\boldsymbol{\alpha}_F^H(k)$

12) $\mathbf{G}_{NF}^{(M)}(k+1)=\mathbf{G}_{NF}^{(0)}(k)$

End for k

In the algorithm, the vectors $\mathbf{x}_F^{(m)}(k)$ and $\mathbf{x}_F^{(m)}(k-N_m)$ are defined as $\left[\mu^{0.5}x_m(k-L),x_m(k),\right.$ $\left.\mu^{0.5}\xi\rho_m(k-L),\xi\rho_m(k)\right]$ and $\left[\mu^{0.5}x_m(k-N_m-L),\right.$ $\left.x_m(k-N_m),\mu^{0.5}\xi\rho_m(k-N_m-L),\xi\rho_m(k-N_m)\right]$. The vectors $\boldsymbol{\alpha}_F^{f(m)}(k)$, $\mathbf{e}_F^{f(m)}(k)$, $\boldsymbol{\alpha}_F^{b(m)}(k)$, $\mathbf{q}_F^{f(m)}(k)$ and $\mathbf{q}_F^{(m)}(k)$ are row-vectors. The matrices $\mathbf{X}_{NF}^{(m)}(k)$ are composed as $\mathbf{X}_{NF}^{(m)}(k) = \left[\mu^{0.5}\boldsymbol{\chi}_N^{(m)}(k-L), \boldsymbol{\chi}_N^{(m)}(k), \mu^{0.5}\xi \times \times \boldsymbol{\rho}_N^{(m)}(k-L), \xi\boldsymbol{\rho}_N^{(m)}(k)\right]$. The columns of the matrix are composed from the vectors. Vectors $\boldsymbol{\chi}_N^{(m)}(k)$ are composed as

$$\boldsymbol{\chi}_N^{(0)}(k) = \boldsymbol{\chi}_N(k),$$
$$\boldsymbol{\chi}_N^{(1)}(k) = \left[\mathbf{x}_{N_1}^T(k-1), \mathbf{x}_{N_2}^T(k), \ldots, \mathbf{x}_{N_m}^T(k), \ldots, \mathbf{x}_{N_M}^T(k)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\chi}_N^{(m)}(k) = \left[\mathbf{x}_{N_1}^T(k-1), \mathbf{x}_{N_2}^T(k-1), \ldots, \mathbf{x}_{N_m}^T(k-1), \mathbf{x}_{N_{m+1}}^T(k), \ldots, \mathbf{x}_{N_M}^T(k)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\chi}_N^{(M)}(k) = \left[\mathbf{x}_{N_1}^T(k-1), \mathbf{x}_{N_2}^T(k-1), \ldots, \mathbf{x}_{N_m}^T(k-1), \ldots, \mathbf{x}_{N_M}^T(k-1)\right]^T,$$

vectors $\boldsymbol{\chi}_N^{(m)}(k-L)$ are composed as

$$\boldsymbol{\chi}_N^{(0)}(k-L) = \boldsymbol{\chi}_N(k-L),$$
$$\boldsymbol{\chi}_N^{(1)}(k-L) = \left[\mathbf{x}_{N_1}^T(k-L-1), \mathbf{x}_{N_2}^T(k-L), \ldots, \mathbf{x}_{N_m}^T(k-L), \ldots, \mathbf{x}_{N_M}^T(k-L)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\chi}_N^{(m)}(k-L) = \left[\mathbf{x}_{N_1}^T(k-L-1), \mathbf{x}_{N_2}^T(k-L-1), \ldots, \mathbf{x}_{N_m}^T(k-L-1), \mathbf{x}_{N_{m+1}}^T(k-L), \ldots, \mathbf{x}_{N_M}^T(k-L)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\chi}_N^{(M)}(k-L) = \left[\mathbf{x}_{N_1}^T(k-L-1), \mathbf{x}_{N_2}^T(k-L-1), \ldots, \mathbf{x}_{N_m}^T(k-L-1), \ldots, \mathbf{x}_{N_M}^T(k-L-1)\right]^T$$

vectors $\boldsymbol{\rho}_N^{(m)}(k)$ are composed as

$$\boldsymbol{\rho}_N^{(0)}(k) = \mathbf{p}_N(k),$$
$$\boldsymbol{\rho}_N^{(1)}(k) = \left[\mathbf{p}_{N_1}^T(k-1), \mathbf{p}_{N_2}^T(k), \ldots, \mathbf{p}_{N_m}^T(k), \ldots, \mathbf{p}_{N_M}^T(k)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\rho}_N^{(m)}(k) = \left[\mathbf{p}_{N_1}^T(k-1), \mathbf{p}_{N_2}^T(k-1), \ldots, \mathbf{p}_{N_m}^T(k-1), \mathbf{p}_{N_{m+1}}^T(k), \ldots, \mathbf{p}_{N_M}^T(k)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\rho}_N^{(M)}(k) = \left[\mathbf{p}_{N_1}^T(k-1), \mathbf{p}_{N_2}^T(k-1), \ldots, \mathbf{p}_{N_m}^T(k-1), \ldots, \mathbf{p}_{N_M}^T(k-1)\right]^T,$$

and vectors $\boldsymbol{\rho}_N^{(m)}(k-L)$ are composed as

$$\boldsymbol{\rho}_N^{(0)}(k-L) = \boldsymbol{\rho}_N(k-L),$$
$$\boldsymbol{\rho}_N^{(1)}(k-L) = \left[\mathbf{p}_{N_1}^T(k-L-1), \mathbf{p}_{N_2}^T(k-L), \ldots, \mathbf{p}_{N_m}^T(k-L), \ldots, \mathbf{p}_{N_M}^T(k-L)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\rho}_N^{(m)}(k-L) = \left[\mathbf{p}_{N_1}^T(k-L-1), \mathbf{p}_{N_2}^T(k-L-1), \ldots, \mathbf{p}_{N_m}^T(k-L-1), \mathbf{p}_{N_{m+1}}^T(k-L), \ldots, \mathbf{p}_{N_M}^T(k-L)\right]^T,$$
$$\vdots$$
$$\boldsymbol{\rho}_N^{(M)}(k-L) = \left[\mathbf{p}_{N_1}^T(k-L-1), \mathbf{p}_{N_2}^T(k-L-1), \ldots, \mathbf{p}_{N_m}^T(k-L-1), \ldots, \mathbf{p}_{N_M}^T(k-L-1)\right]^T.$$

Permutation matrices $\mathbf{S}_{N+1}^{(m)}$ and $\mathbf{T}_{N+1}^{(m)T}$ enable the building of the multichannel fast RLS algorithms with unequal number of weights in channels [16]. The using of the matrices does not require the additional arithmetic operations. The matrices just rearrange Kalman gains. To establish the rearranging rules for the given M and N_m, the matrices product can be calculated in advance. Here, the matrices are for the case M=3:

$$\mathbf{T}_{N+1}^{(1)T} = \begin{bmatrix} 1 & \mathbf{0}_{N_1}^T & \mathbf{0}_{N_2}^T & \mathbf{0}_{N_3}^T \\ \mathbf{0}_{N_1} & \mathbf{I}_{N_1} & \mathbf{O}_{N_1N_2} & \mathbf{O}_{N_1N_3} \\ \mathbf{0}_{N_2} & \mathbf{O}_{N_2N_1} & \mathbf{I}_{N_2} & \mathbf{O}_{N_2N_3} \\ \mathbf{0}_{N_3} & \mathbf{O}_{N_3N_1} & \mathbf{O}_{N_3N_2} & \mathbf{I}_{N_3} \end{bmatrix},$$

$$\mathbf{T}_{N+1}^{(2)T} = \begin{bmatrix} \mathbf{0}_{N_1} & \mathbf{I}_{N_1} & \mathbf{O}_{N_1N_2} & \mathbf{O}_{N_1N_3} \\ 1 & \mathbf{0}_{N_1}^T & \mathbf{0}_{N_2}^T & \mathbf{0}_{N_3}^T \\ \mathbf{0}_{N_2} & \mathbf{O}_{N_2N_1} & \mathbf{I}_{N_2} & \mathbf{O}_{N_2N_3} \\ \mathbf{0}_{N_3} & \mathbf{O}_{N_3N_1} & \mathbf{O}_{N_3N_2} & \mathbf{I}_{N_3} \end{bmatrix},$$

$$\mathbf{T}_{N+1}^{(3)T} = \begin{bmatrix} \mathbf{0}_{N_1} & \mathbf{I}_{N_1} & \mathbf{O}_{N_1N_2} & \mathbf{O}_{N_1N_3} \\ \mathbf{0}_{N_2} & \mathbf{O}_{N_2N_1} & \mathbf{I}_{N_2} & \mathbf{O}_{N_2N_3} \\ 1 & \mathbf{0}_{N_1}^T & \mathbf{0}_{N_2}^T & \mathbf{0}_{N_3}^T \\ \mathbf{0}_{N_3} & \mathbf{O}_{N_3N_1} & \mathbf{O}_{N_3N_2} & \mathbf{I}_{N_3} \end{bmatrix},$$

$$\mathbf{S}_{N+1}^{(1)} = \begin{bmatrix} \mathbf{I}_{N_1} & \mathbf{0}_{N_1} & \mathbf{O}_{N_1N_2} & \mathbf{O}_{N_1N_3} \\ \mathbf{O}_{N_2N_1} & \mathbf{0}_{N_2} & \mathbf{I}_{N_2} & \mathbf{O}_{N_2N_3} \\ \mathbf{O}_{N_3N_1} & \mathbf{0}_{N_3} & \mathbf{O}_{N_3N_2} & \mathbf{I}_{N_3} \\ \mathbf{0}_{N_1}^T & 1 & \mathbf{0}_{N_2}^T & \mathbf{0}_{N_3}^T \end{bmatrix},$$

$$\mathbf{S}_{N+1}^{(1)} = \begin{bmatrix} \mathbf{I}_{N_1} & \mathbf{0}_{N_1} & \mathbf{O}_{N_1N_2} & \mathbf{O}_{N_1N_3} \\ \mathbf{O}_{N_2N_1} & \mathbf{0}_{N_2} & \mathbf{I}_{N_2} & \mathbf{O}_{N_2N_3} \\ \mathbf{O}_{N_3N_1} & \mathbf{0}_{N_3} & \mathbf{O}_{N_3N_2} & \mathbf{I}_{N_3} \\ \mathbf{0}_{N_1}^T & 1 & \mathbf{0}_{N_2}^T & \mathbf{0}_{N_3}^T \end{bmatrix},$$

$$\mathbf{S}_{N+1}^{(3)} = \begin{bmatrix} \mathbf{I}_{N_1} & \mathbf{O}_{N_1N_2} & \mathbf{O}_{N_1N_3} & \mathbf{0}_{N_1} \\ \mathbf{O}_{N_2N_1} & \mathbf{I}_{N_2} & \mathbf{O}_{N_2N_3} & \mathbf{0}_{N_2} \\ \mathbf{O}_{N_3N_1} & \mathbf{O}_{N_3N_2} & \mathbf{I}_{N_3} & \mathbf{0}_{N_3} \\ \mathbf{0}_{N_1}^T & \mathbf{0}_{N_2}^T & \mathbf{0}_{N_3}^T & 1 \end{bmatrix}.$$

The matrices for other values of M can be created similarly.

A parallel form of Fast Transversal Filter (FTF) is based on the recursive updating of the matrices

$$\overline{\mathbf{\Phi}}_F^{(m)}(k) = \mathbf{\Phi}_F^{(m)}(k)\left[\mathbf{I}_F - \frac{\boldsymbol{\alpha}_F^{f(m)H}(k)\mathbf{e}_F^{f(m)}(k)}{E_N^{f(m)}(k)}\right]$$

and

$$\mathbf{\Phi}_F^{(m-1)}(k) = \left[\mathbf{I}_F - \lambda^{-1}\overline{\mathbf{\Phi}}_F^{(m)}(k)\mathbf{q}_F^{(m)H}(k) \times \times \boldsymbol{\alpha}_F^{b(m)}(k)\right]^{-1}\widetilde{\mathbf{\Phi}}_F^{(m)}(k).$$

The matrices allow the calculations of Kalman gains

$$\mathbf{G}_{NF}(k) = \mathbf{G}_{NF}^{(0)}(k) = \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(0)}(k)\mathbf{\Phi}_F^{(0)}(k),$$

and the vectors $\mathbf{e}_F^{f(m)}(k)$ and $\mathbf{e}_F^{b(m)}(k)$, which are used in the updating of prediction error energies

$$E_N^{f(m)}(k) = \lambda E_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k)\boldsymbol{\alpha}_F^{f(m)H}(k),$$

and

$$E_N^{b(m)}(k) = \lambda E_N^{b(m)}(k-1) + \mathbf{e}_F^{b(m)}(k)\boldsymbol{\alpha}_F^{b(m)H}(k).$$

The parallel FTF algorithm is described as:

0) **Init.:** $\boldsymbol{\chi}_N(0) = \mathbf{0}_N, ..., \boldsymbol{\chi}_N(0-L+1) = \mathbf{0}_N$, $\boldsymbol{\rho}_N(0) = \mathbf{0}_N, ..., \boldsymbol{\rho}_N(0-L+1) = \mathbf{0}_N, d(0) = 0, ...,$ $d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF}, \mathbf{h}_N(0) = \mathbf{0}_N$, $E_N^{f(m)}(0) = \delta^2, E_N^{b(m)}(0) = \delta^2\lambda^{-N_m}$, $\mathbf{h}_N^{f(m)}(0) = \mathbf{0}_N, \mathbf{h}_N^{b(m)}(0) = \mathbf{0}_N, m = 1,2,\ldots,M$, $\widetilde{\mathbf{T}}_{NF}^{(M)}(1) = \mathbf{O}_{NF}, \mathbf{\Phi}_F^{(M)}(1) = \mathbf{S}_F$

For $k = 1,2,\ldots,K$

For $m = M, M-1,\ldots,1$

1) $\boldsymbol{\alpha}_F^{f(m)}(k) = \mathbf{x}_F^{(m)}(k) - \mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$

2) $\mathbf{e}_F^{f(m)}(k) = \boldsymbol{\alpha}_F^{f(m)}(k)\mathbf{\Phi}_F^{(m)}(k)$

3) $\boldsymbol{\eta}_F^{(m)}(k) = \boldsymbol{\alpha}_F^{f(m)}(k) / E_N^{f(m)}(k-1)$

4) $\widetilde{\mathbf{T}}_{(N+1)F}^{(m)}(k) = \left[1, \ -\mathbf{h}_N^{f(m)H}(k)\right]^H \boldsymbol{\eta}_F^{(m)}(k) + \left[\mathbf{0}_F^T, \ \widetilde{\mathbf{T}}_{NF}^{(m)}(k)\right]^H$

5) $\mathbf{S}_{N+1}^{(m)}\mathbf{T}_{N+1}^{(m)T}\left\{\widetilde{\mathbf{T}}_{(N+1)F}^{(m)}(k)\right\} = \begin{bmatrix}\mathbf{Q}_{NF}^{(m)}(k) \\ \mathbf{q}_F^{(m)}(k)\end{bmatrix}$

6) $\mathbf{h}_N^{f(m)}(k) = \mathbf{h}_N^{f(m)}(k-1) + \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(m)}(k)\mathbf{e}_F^{f(m)H}(k)$

7) $\boldsymbol{\alpha}_F^{b(m)}(k) = \mathbf{q}_F^{(m)}(k)E_N^{b(m)}(k-1)$

8) $\widetilde{\mathbf{T}}_{NF}^{(m-1)}(k) = \mathbf{Q}_{NF}^{(m)}(k) + \mathbf{h}_N^{b(m)}(k-1)\mathbf{q}_F^{(m)}(k)$

9) $\overline{\mathbf{\Phi}}_F^{(m)}(k) = \mathbf{\Phi}_F^{(m)}(k)\left[\mathbf{I}_F - \dfrac{\boldsymbol{\alpha}_F^{f(m)H}(k)\mathbf{e}_F^{f(m)}(k)}{E_N^{f(m)}(k)}\right]$

11) $\mathbf{\Phi}_F^{(m-1)}(k) = \left[\mathbf{I}_F - \lambda^{-1}\overline{\mathbf{\Phi}}_F^{(m)}(k)\mathbf{q}_F^{(m)H}(k) \times \times \boldsymbol{\alpha}_F^{b(m)}(k)\right]^{-1}\widetilde{\mathbf{\Phi}}_F^{(m)}(k)$

10) $E_N^{f(m)}(k) = \lambda E_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k) \times \times \boldsymbol{\alpha}_F^{f(m)H}(k)$

12) $\mathbf{e}_F^{b(m)}(k) = \boldsymbol{\alpha}_F^{b(m)}(k)\mathbf{\Phi}_F^{(m-1)}(k)$

13) $E_N^{b(m)}(k) = \lambda E_N^{b(m)}(k-1) + \mathbf{e}_F^{b(m)}(k)\boldsymbol{\alpha}_F^{b(m)H}(k)$

14) $\mathbf{h}_N^{b(m)}(k) = \mathbf{h}_N^{b(m)}(k-1) + \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(m-1)}(k)\mathbf{e}_F^{b(m)H}(k)$

End for m

15) $\boldsymbol{\alpha}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$

16) $\mathbf{e}_F(k) = \boldsymbol{\alpha}_F(k)\mathbf{\Phi}_F^{(0)}(k)$

17) $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(m-1)}(k)\mathbf{e}_F^H(k)$

18) $\widetilde{\mathbf{T}}_{NF}^{(M)}(k+1) = \widetilde{\mathbf{T}}_{NF}^{(0)}(k), \mathbf{\Phi}_F^{(M)}(k+1) = \mathbf{\Phi}_F^{(0)}(k)$

End for k

A parallel form of Fast a Posteriori Error Sequential Technique (FAEST) algorithm is distinguished from the parallel FTF algorithm by the recursive update of the inverse matrices

$$\left[\overline{\mathbf{\Phi}}_F^{(m)}(k)\right]^{-1} = \left[\mathbf{\Phi}_F^{(m)}(k)\right]^{-1} + \lambda^{-1}\boldsymbol{\alpha}_F^{f(m)H}(k)\boldsymbol{\eta}_F^{(m)}(k),$$

and

$$\left[\mathbf{\Phi}_F^{(m-1)}(k)\right]^{-1} = \left[\overline{\mathbf{\Phi}}_F^{(m)}(k)\right]^{-1} - \lambda^{-1}\boldsymbol{\alpha}_F^{b(m)H}(k)\widetilde{\mathbf{q}}_F^{(m)}(k).$$

Similarly, a parallel form of a stabilized FAEST algorithm can be developed. A multichannel parallel version of the algorithm is shown below. The details of the single channel prototype of the algorithm can be found in [17].

0) **Init.:** $\boldsymbol{\chi}_N(0) = \mathbf{0}_N, ..., \boldsymbol{\chi}_N(0-L+1) = \mathbf{0}_N$, $\boldsymbol{\rho}_N(0) = \mathbf{0}_N, ..., \boldsymbol{\rho}_N(0-L+1) = \mathbf{0}_N, d(0) = 0,$ $..., d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF}, \mathbf{h}_N(0) = \mathbf{0}_N$, $E_N^{f(m)}(0) = \delta^2, E_N^{b(m)}(0) = \delta^2\lambda^{-N_m}$, $\mathbf{h}_N^{f(m)}(0) = \mathbf{0}_N, \mathbf{h}_N^{b(m)}(0) = \mathbf{0}_N, m = 1,2,\ldots,M$, $\widetilde{\mathbf{T}}_{NF}^{(M)}(1) = \mathbf{O}_{NF}, \mathbf{\Phi}_F^{(M)}(1) = \mathbf{S}_F$

For $k = 1,2,\ldots,K$

For $m = M, M-1,\ldots,1$

1) $\boldsymbol{\alpha}_F^{f(m)}(k) = \mathbf{x}_F^{(m)}(k) - \mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$

2) $\mathbf{e}_F^{f(m)}(k) = \boldsymbol{\alpha}_F^{f(m)}(k)\boldsymbol{\Phi}_F^{(m)}(k)$

3) $\boldsymbol{\eta}_F^{(m)}(k) = \boldsymbol{\alpha}_F^{f(m)}(k) / E_N^{f(m)}(k-1)$

4) $\widetilde{\mathbf{T}}_{(N+1)F}^{(m)}(k) = \begin{bmatrix} 1 \\ -\mathbf{h}_N^{f(m)}(k-1) \end{bmatrix} \boldsymbol{\eta}_F^{(m)}(k) + \begin{bmatrix} \mathbf{0}_F^T \\ \widetilde{\mathbf{T}}_{NF}^{(m)}(k) \end{bmatrix}$

5) $\mathbf{S}_{N+1}^{(m)}\mathbf{T}_{N+1}^{(m)T}\left\{\widetilde{\mathbf{T}}_{(N+1)F}^{(m)}(k)\right\} = \begin{bmatrix} \widetilde{\mathbf{Q}}_{NF}^{(m)}(k) \\ \widetilde{\mathbf{q}}_F^{(m)}(k) \end{bmatrix}$

6) $\mathbf{h}_N^{f(m)}(k) = \mathbf{h}_N^{f(m)}(k-1) + \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(m)}(k)\mathbf{e}_F^{f(m)H}(k)$

7) $E_N^{f(m)}(k) = \lambda E_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k)\boldsymbol{\alpha}_F^{f(m)H}(k)$

8) $\left[\overline{\boldsymbol{\Phi}}_F^{(m)}(k)\right]^{-1} = \left[\boldsymbol{\Phi}_F^{(m)}(k)\right]^{-1} + \lambda^{-1}\boldsymbol{\alpha}_F^{f(m)H}(k)\boldsymbol{\eta}_F^{(m)}(k)$

9) $\boldsymbol{\alpha}_F^{b(m)}(k) = \mathbf{x}_F^{(m)}(k-N_m) - \mathbf{h}_N^{b(m)H}(k-1)\mathbf{X}_{NF}^{(m-1)}(k)$

10) $\mathbf{q}_F^{(m)}(k) = \boldsymbol{\alpha}_F^{b(m)}(k) / E_N^{b(m)}(k-1)$

11) $\widetilde{\boldsymbol{\alpha}}_F^{b(m)}(k) = \widetilde{\mathbf{q}}_F^{(m)}(k)E_N^{b(m)}(k-1)$

12) $\boldsymbol{\alpha}_F^{b(1)(m)}(k) = K_1\boldsymbol{\alpha}_F^{b(m)}(k) + (1-K_1)\widetilde{\boldsymbol{\alpha}}_F^{b(m)}(k)$

13) $\boldsymbol{\alpha}_F^{b(2)(m)}(k) = K_2\boldsymbol{\alpha}_F^{b(m)}(k) + (1-K_2)\widetilde{\boldsymbol{\alpha}}_F^{b(m)}(k)$

14) $\boldsymbol{\alpha}_F^{b(5)(m)}(k) = K_5\boldsymbol{\alpha}_F^{b(m)}(k) + (1-K_5)\widetilde{\boldsymbol{\alpha}}_F^{b(m)}(k)$

15) $\mathbf{t}_F^{(m)}(k) = K_4\mathbf{q}_F^{(m)}(k) + (1-K_4)\widetilde{\mathbf{q}}_F^{(m)}(k)$

16) $\widetilde{\mathbf{T}}_{NF}^{(m-1)}(k) = \widetilde{\mathbf{Q}}_{NF}^{(m)}(k) + \mathbf{h}_N^{b(m)}(k-1)\mathbf{t}_F^{(m)}(k)$

17) $\left[\hat{\boldsymbol{\Phi}}_F^{(m-1)}(k)\right]^{-1} = \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(m-1)H}(k)\mathbf{X}_{NF}^{(m-1)}(k) + \mathbf{S}_F$

18) $\left[\widetilde{\boldsymbol{\Phi}}_F^{(m-1)}(k)\right]^{-1} = \left[\overline{\boldsymbol{\Phi}}_F^{(m)}(k)\right]^{-1} - {}$
$- \lambda^{-1}\boldsymbol{\alpha}_F^{b(5)(m)H}(k)\widetilde{\mathbf{q}}_F^{(m)}(k)$

19) $\left[\boldsymbol{\Phi}_F^{(m-1)}(k)\right]^{-1} = K_3\left[\hat{\boldsymbol{\Phi}}_F^{(m-1)}(k)\right]^{-1} + {}$
$+ (1-K_3)\left[\widetilde{\boldsymbol{\Phi}}_F^{(m-1)}(k)\right]^{-1}$

20) $\mathbf{e}_F^{b(1)(m)}(k) = \boldsymbol{\alpha}_F^{b(1)(m)}(k)\widetilde{\boldsymbol{\Phi}}_F^{(m-1)}(k)$

21) $\mathbf{e}_F^{b(2)(m)}(k) = \boldsymbol{\alpha}_F^{b(2)(m)}(k)\widetilde{\boldsymbol{\Phi}}_F^{(m-1)}(k)$

22) $E_N^{b(m)}(k) = \lambda E_N^{b(m)}(k-1) + \mathbf{e}_F^{b(2)(m)}(k)\boldsymbol{\alpha}_F^{b(2)(m)H}(k)$

23) $\mathbf{h}_N^{b(m)}(k) = \mathbf{h}_N^{b(m)}(k-1) + \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(m-1)}(k)\mathbf{e}_F^{b(1)(m)H}(k)$

End for m

24) $\boldsymbol{\alpha}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$

25) $\mathbf{e}_F(k) = \boldsymbol{\alpha}_F(k)\boldsymbol{\Phi}_F^{(0)}(k)$

25) $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + \lambda^{-1}\widetilde{\mathbf{T}}_{NF}^{(0)}(k)\mathbf{e}_F^H(k)$

25) $\widetilde{\mathbf{T}}_{NF}^{(M)}(k+1) = \widetilde{\mathbf{T}}_{NF}^{(0)}(k), \boldsymbol{\Phi}_F^{(M)}(k+1) = \boldsymbol{\Phi}_F^{(0)}(k)$

End for k

It is well known, that RLS and fast RLS adaptive filtering algorithms can be also developed by the using QRD. If the adaptive filter weights are required, the inverse QRD has to be used. Usually, QRD RLS algorithms have the square root operations. The operations can be excludes by means the scaling of the variables involved in calculations [18]. Using the duality between the fast RLS algorithms and the fast QRD based least squares algorithms [19], parallel multichannel version of the square root free inverse QRD fast RLS algorithm [20] can be developed. The algorithm is presented below.

0) **Init.:** $\boldsymbol{\chi}_N(0) = \mathbf{0}_N, \ldots, \boldsymbol{\chi}_N(0-L+1) = \mathbf{0}_N$, $\boldsymbol{\rho}_N(0) = \mathbf{0}_N, \ldots, \boldsymbol{\rho}_N(0-L+1) = \mathbf{0}_N, d(0) = 0, \ldots,$ $d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF}, \mathbf{h}_N(0) = \mathbf{0}_N$, $E_N^{f(m)}(0) = \delta^2, E_N^{b(m)}(0) = \delta^2\lambda^{-N_m}, \mathbf{h}_N^{f(m)}(0) = \mathbf{0}_N$, $\mathbf{h}_N^{b(m)}(0) = \mathbf{0}_N, m = 1,2,\ldots,M, \mathbf{G}_{NF}^{(M)}(1) = \mathbf{O}_{NF}$, $\mathbf{K}_F^{B(M)}(1) = \mathbf{S}_F$

For $k = 1,2,\ldots,K$

For $m = M, M-1,\ldots,1$

1) $\boldsymbol{\alpha}_F^{f(m)}(k) = \mathbf{x}_F^{(m)}(k) - \mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$

2) $\mathbf{e}_F^{f(m)}(k) = \boldsymbol{\alpha}_F^{f(m)}(k)\left[\mathbf{K}_F^{B(m)}(k)\right]^{-1}$

3) $\overline{\mathbf{K}}_F^{B(m)}(k) = \mathbf{K}_F^{B(m)}(k) + \lambda^{-1}E_N^{-1f(m)}(k-1) \times$
$\times \boldsymbol{\alpha}_F^{f(m)H}(k)\boldsymbol{\alpha}_F^{f(m)}(k)$

4) $\mathbf{C}_F^{f(m)}(k) = \mathbf{K}_F^{B(m)}(k)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}$

5) $\mathbf{s}_F^{f(m)}(k) = \lambda^{-1}E_N^{-1f(m)}(k-1)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}\boldsymbol{\alpha}_F^{f(m)H}(k)$

6) $\mathbf{Q}_{NF}^{f(m)}(k) = \mathbf{G}_{NF}^{(m)}(k)\mathbf{C}_F^{f(m)}(k) -$
$- \mathbf{h}_N^{f(m)}(k-1)\mathbf{s}_F^{f(m)H}(k)$ or 8)

7) $\mathbf{h}_N^{f(m)}(k) = \mathbf{h}_N^{f(m)}(k-1) + \mathbf{G}_{NF}^{(m)}(k)\boldsymbol{\alpha}_F^{f(m)H}(k)$

8) $\mathbf{Q}_{NF}^{f(m)}(k) = \mathbf{G}_{NF}^{(m)}(k) - \mathbf{h}_N^{f(m)}(k)\mathbf{s}_F^{f(m)H}(k)$ or 6)

9) $\mathbf{q}_F^{f(m)}(k) = \mathbf{s}_F^{f(m)H}(k)$

10) $\mathbf{S}_{N+1}^{(m)}\mathbf{T}_{N+1}^{(m)T}\begin{bmatrix} \mathbf{q}_F^{f(m)}(k) \\ \mathbf{Q}_{NF}^{f(m)}(k) \end{bmatrix} = \begin{bmatrix} \mathbf{Q}_{NF}^{b(m)}(k) \\ \mathbf{q}_F^{b(m)}(k) \end{bmatrix}$

11) $E_N^{f(m)}(k) = \lambda E_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k)\boldsymbol{\alpha}_F^{f(m)H}(k)$

12) $\boldsymbol{\alpha}_F^{b(m)}(k) = \mathbf{x}_F^{(m)}(k-N_m) - \mathbf{h}_N^{b(m)H}(k-1)\mathbf{X}_{NF}^{(m-1)}(k)$

13) $\mathbf{K}_F^{B(m-1)}(k) = \overline{\mathbf{K}}_F^{B(m)}(k) - \lambda^{-1}E_N^{-1b(m)}(k-1) \times \times \boldsymbol{\alpha}_F^{b(m)H}(k)\boldsymbol{\alpha}_F^{b(m)}(k)$

14) $\mathbf{C}_F^{b(m)}(k) = \mathbf{K}_F^{B(m-1)}(k)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}$

15) $\mathbf{s}_F^{b(m)}(k) = \lambda^{-1}E_N^{-1b(m)}(k-1)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}\boldsymbol{\alpha}_F^{b(m)H}(k)$

16) $\mathbf{G}_{NF}^{(m-1)}(k) = \left[\mathbf{Q}_{NF}^{b(m)}(k) + \mathbf{h}_N^{b(m)}(k-1)\mathbf{s}_F^{b(m)H}(k)\right] \times \times \left[\mathbf{C}_F^{b(m)}(k)\right]^{-1}$

17) $\mathbf{e}_F^{b(m)}(k) = \boldsymbol{\alpha}_F^{b(m)}(k)\left[\mathbf{K}_F^{B(m-1)}(k)\right]^{-1}$

18) $E_N^{b(m)}(k) = \lambda E_N^{b(m)}(k-1) + \mathbf{e}_F^{b(m)}(k)\boldsymbol{\alpha}_F^{b(m)H}(k)$

19) $\mathbf{h}_N^{b(m)}(k) = \mathbf{h}_N^{b(m)}(k-1) + \mathbf{G}_{NF}^{(m-1)}(k)\boldsymbol{\alpha}_F^{b(m)H}(k)$

End for m

20) $\boldsymbol{\alpha}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$

21) $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + \mathbf{G}_{NF}^{(0)}(k)\boldsymbol{\alpha}_F^H(k)$

22) $\mathbf{G}_{NF}^{(M)}(k+1) = \mathbf{G}_{NF}^{(0)}(k), \mathbf{K}_F^{B(M)}(k+1) = \mathbf{K}_F^{B(0)}(k)$

End for k

Similarly, the stabilized form of the parallel inverse QRD-based algorithm can be developed as well.

The identity of the parallel and the same named sequential fast RLS algorithms is ensured by means of the square matrices $\left[\boldsymbol{\Phi}_F^{(m)}(k)\right]^{-1}$ and $\mathbf{K}_F^{B(m)}(k)$ which have $F \times F$ elements. The matrices disappear in sequential SW regularized fast RLS algorithms. They become the scalar variables, known in adaptive filter theory as the inverse of likehood ratios.

The above considered parallel algorithms can be used in adaptive filters without linear constraints and for the calculation of Kalman gain in LC RLS algorithms. In the parallel LC RLS algorithms, MIL (6) is also used for the calculation of the matrices $\boldsymbol{\Gamma}_{NJ}(k) = \mathbf{R}_N^{-1}(k)\mathbf{C}_{NJ}$, $\boldsymbol{\Psi}_J^{-1}(k) = \left[\mathbf{C}_{NJ}^H\boldsymbol{\Gamma}_{NJ}(k)\right]^{-1}$ and $\mathbf{Q}_{NJ}(k) = \boldsymbol{\Gamma}_{NJ}(k)\boldsymbol{\Psi}_J^{-1}(k)$, caused by linear constrains in (2). There are three forms of the LC RLS algorithms [13]. The parallel forms of the algorithms are developed by the using of steps similar to those considered in [13].

The below is LC RLS algorithm, based on the matrix $\mathbf{Q}_{NJ}(k)$ calculation. The algorithms is parallel as the computation of the matrices with F columns can be accomplished by means of F parallel processors, because the computations are independent each other and depend on the independent streams of adaptive filters input data, i.e. the columns of matrix $\mathbf{X}_{NF}(k)$.

0) **Init.:** $\boldsymbol{\chi}_N(0) = \mathbf{0}_N, \ldots, \boldsymbol{\chi}_N(0-L+1) = \mathbf{0}_N$,
$\boldsymbol{\rho}_N(0) = \mathbf{0}_N, \ldots, \boldsymbol{\rho}_N(0-L+1) = \mathbf{0}_N$,
$d(0) = 0, \ldots, d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF}$,
$\mathbf{R}_N^{-1}(0) = \delta^{-2}\boldsymbol{\Lambda}_N, \boldsymbol{\Gamma}_{NJ}(0) = \mathbf{R}_N^{-1}(0)\mathbf{C}_{NJ}$,
$\mathbf{Q}_{NJ}(0) = \boldsymbol{\Gamma}_{NJ}(0)\left[\mathbf{C}_{NJ}^H\boldsymbol{\Gamma}_{NJ}(0)\right]^{-1}, \mathbf{h}_N(0) = \mathbf{Q}_{NJ}(0)\mathbf{f}_J$
$\boldsymbol{\Lambda}_N = diag(1, \lambda, \ldots, \lambda^{N_1-1}, \ldots, 1, \lambda, \ldots, \lambda^{N_M-1})$

For $k = 1, 2, \ldots, K$

1) Calculation of $\mathbf{G}_{NF}(k)$

2) $\mathbf{V}_{JF}(k) = \mathbf{C}_{NJ}^H\mathbf{G}_{NF}(k)$

3) $\mathbf{N}_{JF}^H(k) = \mathbf{X}_{NF}^H(k)\mathbf{Q}_{NJ}(k-1)$

4) $\mathbf{Q}'_{NJ}(k) = \left[\mathbf{Q}_{NJ}(k-1) - \mathbf{G}_{NF}(k)\mathbf{N}_{JF}^H(k)\right] \times \times \left[\mathbf{I}_J + \dfrac{\mathbf{V}_{JF}(k)\mathbf{N}_{JF}^H(k)}{\mathbf{I}_F - \mathbf{N}_{JF}^H(k)\mathbf{V}_{JF}(k)}\right]$

5) $\mathbf{Q}_{NJ}(k) = \mathbf{Q}'_{NJ}(k) + \mathbf{C}_{NJ}\left(\mathbf{C}_{NJ}^H\mathbf{C}_{NJ}\right)^{-1} \times \times \left[\mathbf{I}_J - \mathbf{C}_{NJ}^H\mathbf{Q}'_{NJ}(k)\right]$

6) $\boldsymbol{\alpha}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$

7) $\mathbf{h}'_N(k) = \mathbf{h}_N(k-1) + \mathbf{G}_{NF}(k)\boldsymbol{\alpha}_F^H(k)$

8) $\mathbf{h}_N(k) = \mathbf{h}'_N(k) + \mathbf{Q}_{NJ}(k)\left[\mathbf{f}_J - \mathbf{C}_{NJ}^H\mathbf{h}'_N(k)\right]$

End for k

4. Simulation

Simulations, which confirm the considered parallel algorithms efficiency, are presented in the section in Fig. 2 – Fig. 5. Multichannel versions of PW LC RLS and unregularized SW LC RLS algorithms are compared in a three channel linear filter identification problem and non-stationary (speech) input signals. Filter performance is observed during 50000 iterations, corresponding to the same number of signal samples. In Fig. 2 and Fig. 4, the vertical arrows indicate the mentioned frequencies and horizontal dashed line indicates the level of constraints.

Fig. 2 demonstrates that two linear constraints 0 dB of adaptive filter transfer function $|H(f)|$, applied at f=1 kHz and f=2 kHz selected frequency points, are provided by the both algorithms. The Echo Return Loss Enhancement (ERLE) is frequently used in the evaluation of the quality of an identification problem solution. The ERLE is the ratio of the energy of the echo signal $d(k)$ and the energy of the suppressed echo $\alpha_{N,\chi}(k)$, measured at each iteration of adaptive filtering algorithm as follows:

$$ERLE(k) = 10\log_{10}\left(\sum_{i=k-B+1}^{k} d^2(i) \Big/ \sum_{i=k-B+1}^{k} \alpha_{N,\chi}^2(i) \right)$$

Here, the length B is approximately selected as 30 ms that corresponds to a stationarity interval of speech. From Fig. 3, it can be seen that the performance of the SW algorithm in terms of ERLE is better, meaning less error at the adaptive filter output in comparison with PW ones.

It is interesting to examine the behavior of the algorithms as presented in Fig. 3. At the beginning of the adaptation process, both algorithms provide approximately similar performance. However, the PW algorithm does not properly track the changing processed signals. At the same time, the SW algorithm demonstrates better performance (approximately 20 dB higher ERLE in the considered case) due to its ability to track the change in the input signals. In the examples, L=256 samples or approximately 30 ms. The window length for ERLE calculation is also 256 samples.

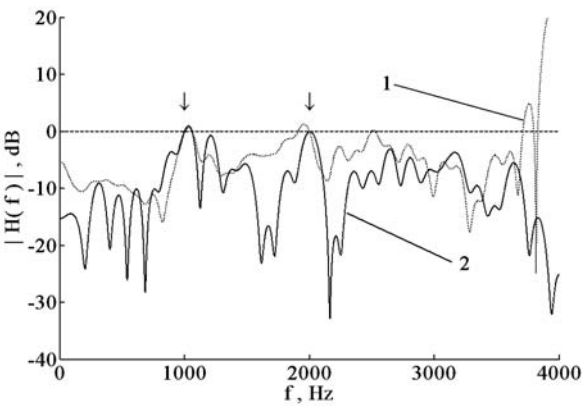

Fig. 2. Simulations results, transfer function: 1 - PW LC RLS algorithm without regularization, 2 - SW LC RLS algorithm without regularization.

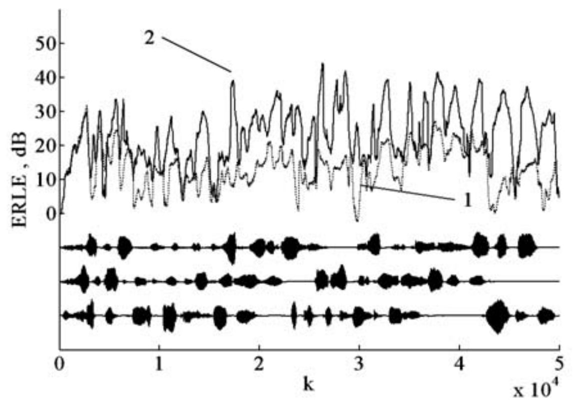

Fig. 3. Simulations results, ERLE: 1 - PW LC RLS algorithm without regularization, 2 - SW LC RLS algorithm without regularization.

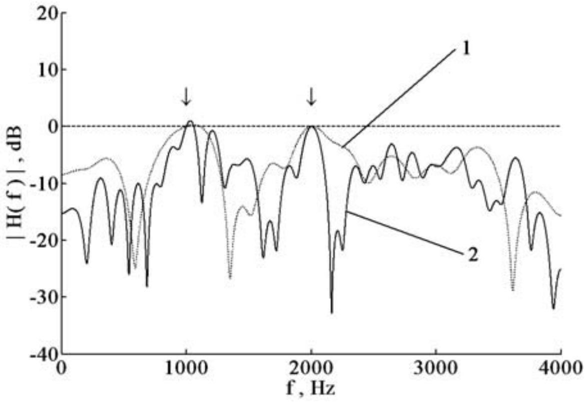

Fig. 4. Simulations results, transfer function: 1 - SW LC RLS algorithm with regularization, 2 - SW LC RLS algorithm without regularization.

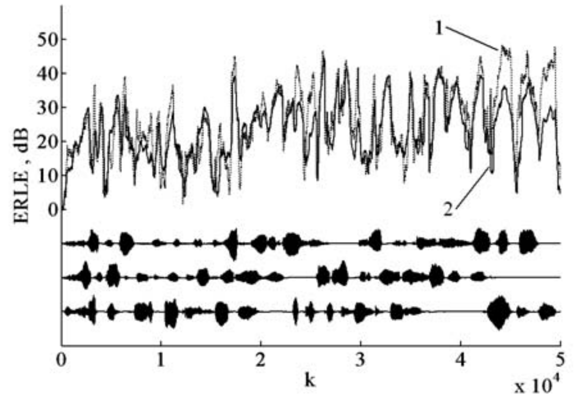

Fig. 5. Simulations results, ERLE: 1 - SW LC RLS algorithm with regularization, 2 - SW LC RLS algorithm without regularization.

The improvement of the SW RLS algorithms is achieved by means of dynamic regularization of correlation matrix, see Fig. 4 and Fig. 5. The regularized algorithm also provides the mentioned constraints of adaptive filter transfer function. Besides, average ERLE in the case is compared with unregularized algorithm and even higher. It means more stable performance of the regularized algorithms in comparing with unregularized ones.

So, the above results confirm the operation of the proposed algorithms. Better efficiency of constrained SW RLS algorithms was also demonstrated in comparison with PW algorithms, in situations where the input signals of the adaptive filter are nonstationary. Furthermore, algorithm improvement is achieved due to the regularization of the correlation matrix inversion.

5. Conclusion

Thus, a simple approach to the description of the multichannel parallel RLS algorithms diversity, caused by the possible modifications of correlation matrix, was presented in the paper. The parallel algorithms are mathematically identical to the same named sequential algorithms. Identity means the same performance if adaptive filters have the same parameters and process the same signals. Total number of arithmetic operations of the parallel algorithms and the same named sequential algorithms is approximately the same. However, if F processors are used, the computational load per processor is decreased F times in the parallel algorithms. These algorithms can be used in all traditional applications of adaptive filters.

References

[1] SAYED, A. H. *Fundamentals of Adaptive Filtering*. Hoboken, NJ: John Wiley and Sons, Inc., 2003.

[2] BENESTY, J., HUANG, Y. (Eds.). *Adaptive Signal Processing: Applications to Real-World Problems*. New York: Springer-Verlag, 2003.

[3] DJIGAN, V. I. Multichannel RLS and fast RLS adaptive filtering algorithms. *Successes of Modern Radioelectronics*. 2004, no. 11, p. 48 - 77. (in Russian).

[4] PETRICHKOVICH Y. Y., SOLOKHINA, T. V. MULTICORE signal controllers - new Russian family of SoC. In *Proceedings of 6-th International Conference on Digital Signal Processing and its Applications*. Moscow (Russia), 2004, vol.1, p. 8 - 15. (in Russian).

[5] GAY, S. L. Dynamically regularized fast RLS with application to echo cancellation. In *Proceedings of the International Conference on Acoustic Speech and Signal Processing*. Atlanta (USA), 1996, p. 957 - 960.

[6] PAPAODYSSEYS, C. A robust, parallelizable, O(m), a posteriori recursive least squares algorithm for efficient adaptive filtering. *IEEE Trans. Signal Processing*, 1999, vol. 47, no. 9, p. 2552 - 2558.

[7] DJIGAN, V. I. Parallelizable sliding window regularized fast RLS algorithm for multichannel linearly constrained adaptive filtering. In *Proceedings of the 10-th International Conference on Radars, Navigation and Communication (RLNC-2004)*. Voronezh (Russia), 2004, vol. 1, p. 132 - 142. (in Russian).

[8] DJIGAN, V. I. Multichannel fast RLS adaptive filtering algorithm for parallel implementation by means of four processors. In *Proceedings of Bauman's Moscow State Technical University*, 2005, no. 1, p. 83 - 99. (in Russian).

[9] DJIGAN, V. I. On parallel implementation of adaptive filtering algorithms. In *Proceedings of the 8–th International Conference on Pattern Recognition and Information Processing (PRIP–2005)*. Minsk (Belarus), 2005, vol. 1, p. 101 - 104.

[10] DJIGAN, V. I. Fast RLS with parallel computations. In *Proceedings of the IEEE 7–th Emerging Technologies Workshop: "Circuits and Systems for 4G Mobile Wireless Communications"*. S. Petersburg (Russia), 2005, p. 42 - 45.

[11] DJIGAN, V. I. Parallel multichannel fast RLS adaptive filtering algorithm based on inverse QR decomposition. In *Proceedings of the Second IASTED International Multi-Conference on ACIT*. Novosibirsk (Russia), 2005, vol. SIP, p. 170 – 175.

[12] DJIGAN, V. I. Parallel linearly–constrained recursive least squares for mulitchannel adaptive filtering. In *Proceedings of St. Petersburg IEEE Chapters: International Conference "Radio – That Connects Time. 110 Years of Radio Invention"*. S. Petersburg (Russia), 2005, vol. 2, p. 134 -139.

[13] RESENDE, L. S., ROMANO, J. M. T., BELLANGER, M.G. A fast least-squares algorithm for linearly constrained adaptive filtering. *IEEE Trans. Signal Processing*, 1996, vol. 44, no. 5, p. 1168 - 1174.

[14] GIORDANO, A. A., HSU, F. M. *Least Square Estimation with Application to Digital Signal Processing*. Toronto: John Wiley and Sons, Inc., 1985.

[15] ZELNIKER, G., TAYLOR, F. J. *Advanced Digital Signal Processing: Theory and Applications*. New York: Marcel Dekker, Inc., 1994.

[16] GLENTIS, G.-O. A., KALOUPTSIDIS N. Fast adaptive algorithms for multichannel filtering and system identification. *IEEE Trans. Signal Processing*, 1992, vol. 40, no. 10, p. 2433 - 2458.

[17] SLOCK, D.T.M., KAILATH, T. Numerically stable fast transversal filters for recursive least squares adaptive filtering. *IEEE Trans. Signal Processing*. 1991, vol. 39, no. 1, p. 92 - 114.

[18] HSIEH, S. F., LIU, K. J. R. A unified square-root-free approach for QRD based recursive least squares estimation. *IEEE Trans. Signal Processing,* 1993, vol. 41, no. 3, p. 1405 - 1409.

[19] GLENTIS, G.-O. A. On the duality between the fast transversal and the fast QRD adaptive least squares algorithms. *IEEE Trans. Signal Processing*. 1999, vol. 47, no. 8, p. 2317 - 2321.

[20] PROUDER, I. K. Fast time-series adaptive-filtering algorithm based on the QRD inverse-updates method. *IEE Proceedings: Vision, Image and Signal Processing*, 1994, vol. 141, no. 5, p. 325 - 333.

About Author...

Victor I. DJIGAN was born in 1958 in Ukraine. He received a B.S. degree with honors from the Vinnitsa Technical College of Electronic Devices, Ukraine, in 1978; M.S. degree with honors from the Moscow Institute of Electronic Engineering (MIEE) - Technical University, Russia, in 1984, and Ph.D. degree from MIEE in 1990, all in radio engineering. From 1984 to the present he has been holding different R&D positions in MIEE, in a number of Russian companies and in such international companies as Samsung Electronics, LG and PMC-Sierra outside Russia, where he worked in the fields of digital signal processing; communication; adaptive antenna arrays; acoustic echo and noise cancellation; transmitting line analysis, simulation and reflectometry. Presently, he is a Principal Researcher in ELVEES R&D Center, Moscow, Russia, and a Scientific Secretary of the Company. His scientific interests include adaptive signal processing, speech processing and digital communication. He is the author of about 100 papers in the above-mentioned fields, Member of the Russian A.S. Popov Society for Radioengineering, Electronics & Communications, and IEEE Senior Member (M'96, SM'2004).