Diversity of RLS for parallel implementation

VICTOR I. DJIGAN
VLSI Development Laboratory
ELVEES R&D Center of Microelectronics
POB 19, Centralny Prospect, Zelenograd, Moscow
RUSSIA 124460
http://www.elvees.ru

Abstract: - The paper presents a family of the sliding window RLS adaptive filtering algorithms with regularization of adaptive filter correlation matrix. The algorithms are 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.

Key-Words: - 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 - it is 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.

The simplest gradient adaptive filtering algorithms are basically used in the applications 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, the 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) sliding window is used. In SW case, MIL is used twice per sample. Due to the limited number of samples involved in the estimation of correlation matrix, SW RLS algorithms can be unstable. 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. Such algorithms are fitted to the implementation by means of two or four processors [6-9].

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, as the sequence of the same parallel computations, similar to 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.

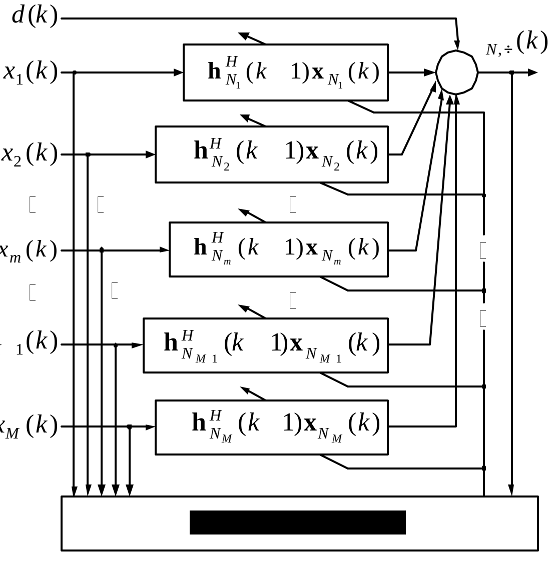
$d(k)$
$x_1(k)$
$x_2(k)$
$x_m(k)$
$x_{M\ 1}(k)$
$x_M(k)$
$\mathbf{h}^H_{N_1}(k\quad 1)\mathbf{x}_{N_1}(k)$
$\mathbf{h}^H_{N_2}(k\quad 1)\mathbf{x}_{N_2}(k)$
$\mathbf{h}^H_{N_m}(k\quad 1)\mathbf{x}_{N_m}(k)$
$\mathbf{h}^H_{N_{M\ 1}}(k\quad 1)\mathbf{x}_{N_{M\ 1}}(k)$
$\mathbf{h}^H_{N_M}(k\quad 1)\mathbf{x}_{N_M}(k)$
$_{N,\div}(k)$
[redacted]

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:

$$E_N(k) = \sum_{i=k-L+1}^{k} l^{k-i}\left[d(i) - \mathbf{h}^H_N(k)\div_N(i)\right]^2 \ , \qquad (1)$$

where the error is measured over an observation window L samples long. The minimization is carried out under the condition $\mathbf{C}^H_{NJ}\mathbf{h}_{N,\div}(k) = \mathbf{f}_J$. Here, $\mathbf{h}^H_N(k) = \left[\mathbf{h}^H_{N_1}(k), \mathbf{h}^H_{N_2}(k), \ldots, \mathbf{h}^H_{N_m}(k), \ldots, \mathbf{h}^H_{N_M}(k)\right]$ 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; $\div^T_N(k) = \left[\mathbf{x}^T_{N_1}(k), \mathbf{x}^T_{N_2}(k), \ldots, \mathbf{x}^T_{N_m}(k), \ldots, \mathbf{x}^T_{N_M}(k)\right]$ is a vector of input signals in the adaptive filter; $\mathbf{x}^T_{N_m}(k) = \left[x_m(k), x_m(k-1), \ldots, x_m(k - N_m + 1)\right]$ is a vector of signals in m -th channel; $\mathbf{C}_{NJ}$ and $\mathbf{f}_J$ are matrix and vector of 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 l is a forgetting factor. Superscripts H è T denote Hermitian transpose and transposition of a vector or a matrix; one subscript N , J or F denote 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 [10]:

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

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

$$\begin{aligned}\mathbf{R}_N(k) &= \sum_{i=k-L+1}^{k} l^{k-i}\left[\div_N(i)\div^H_N(i) + x^2\tilde{\mathbf{n}}_N(i)\tilde{\mathbf{n}}^T_N(i)\right] = \\ &= l\mathbf{R}_N(k-1) + \div_N(k)\div^H_N(k) - m\div_N(k-L)\times \\ &\times \div^H_N(k-L) + x^2\tilde{\mathbf{n}}_N(k)\tilde{\mathbf{n}}^T_N(k) - mx^2\tilde{\mathbf{n}}_N(k-L)\times \\ &\times \tilde{\mathbf{n}}^T_N(k-L)\end{aligned} \qquad (3)$$

The crosscorrelation of $\div_N(k)$ and $d(k)$ is defined as

$$\begin{aligned}\mathbf{r}_N(k) &= \sum_{i=k-L+1}^{k} l^{k-i}\div_N(i)d^*(i) = l\mathbf{r}_N(k-1) + \\ &+ \div_N(k)d^*(k) - m\div_N(k-L)d^*(k-L)\end{aligned} \qquad (4)$$

In (3) and (4), $(\ )^*$ means complex conjugate, $m = l^L$ and x^2 is a small value of a dynamic regularization parameter [5]. Parameter x^2 and parameter d^2 for the initial regularization of correlation matrix are selected as $x^2, d^2 \geq 0.01s^2_x$, where s^2_x is the variance of adaptive filter input signals. The dynamic regularization vector $\tilde{\mathbf{n}}_N(k)$ is defined as

$$\tilde{\mathbf{n}}^T_N(k) = \left[\mathbf{p}^T_{N_1}(k), \mathbf{p}^T_{N_2}(k), \ldots, \mathbf{p}^T_{N_m}(k), \ldots, \mathbf{p}^T_{N_M}(k)\right], \ (5)$$

where $\mathbf{p}^T_{N_m}(k) = \left[p_m(k), p_m(k-1), \ldots, p_m(k - N_m + 1)\right]$ 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.

The sequential application of MIL [1] to equation (3) allows the getting of sequential RLS algorithms, and the application of the MIL [6] allows the getting of parallel RLS algorithms. The MIL [6] is very cumbersome and the resulting mathematical descriptions of parallel algorithms [6-9] are cumbersome as well.

In general case, see [11], 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 (5) is the key tool for the development of the parallel SW regularized RLS algorithms, considered in this paper. To use (6), the matrices $\mathbf{C} = [\mathbf{y}, \mathbf{x}, \mathbf{z}, \mathbf{v}] = \left[m^{0.5}\div_N(k-L), \div_N(k), m^{0.5}x\tilde{\mathbf{n}}_N(k-L), x\tilde{\mathbf{n}}_N(k)\right]$, $\mathbf{D} = \mathbf{C}^H$ 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, see below.

```
    Init.: $\div_N(0) = \mathbf{0}_N, \ldots, \div_N(0-L+1) = \mathbf{0}_N,$
    $\tilde{\mathbf{n}}_N(0) = \mathbf{0}_N, \ldots, \tilde{\mathbf{n}}_N(0-L+1) = \mathbf{0}_N,$
0)  $d(0) = 0, \ldots, d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF},$
    $\mathbf{R}_N^{-1}(0) = d^{-2}\ddot{\mathbf{E}}_N, \mathbf{h}_N(0) = \mathbf{0}_N$
    $\ddot{\mathbf{E}}_N = diag(1, l, \ldots, l^{N_1-1}, \ldots, 1, l, \ldots, l^{N_M-1})$
For  $k = 1, 2, \ldots, K$
1)  $\mathbf{G}_{NF}(k) = \dfrac{\mathbf{R}_N^{-1}(k-1)\mathbf{X}_{NF}(k)}{l\mathbf{S}_F + \mathbf{X}_{NF}^H(k)\mathbf{R}_N^{-1}(k-1)\mathbf{X}_{NF}(k)}$
2)  $\mathbf{R}_N^{-1}(k) = l^{-1}\left[\mathbf{R}_N^{-1}(k-1) - \mathbf{G}_{NF}(k) \times\right.$
    $\left.\times \mathbf{X}_{NF}^H(k)\mathbf{R}_N^{-1}(k-1)\right]$
3)  $\acute{\mathbf{a}}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$
4)  $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + \mathbf{G}_{NF}(k)\acute{\mathbf{a}}_F^H(k)$
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 $\left[m^{0.5}d(k-L), d(k), 0, 0\right]$, and error signal at the output of adaptive filter, see Fig. 1, is defined as $\boldsymbol{a}_{N,\div}(k) = d(k) - \mathbf{h}_N^H(k-1)\div_N(k) = \boldsymbol{a}_F^{(2)}(k)$, where $a_F^{(2)}(k)$ means the second element of the error vector $\acute{\mathbf{a}}_F(k)$. Vectors $\mathbf{d}_F(k)$ and $\acute{\mathbf{a}}_F(k)$ 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 $F \times F$ elements in the parallel algorithm. The matrix ensures the mathematic identity of the sequential and parallel RLS algorithms. Because $F \le 4$, the matrix inversion does not effect on 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 [12]. A parallel version of the Fast Kalman (FK) algorithm is shown below.

```
    Init.: $\div_N(0) = \mathbf{0}_N, \ldots, \div_N(0-L+1) = \mathbf{0}_N,$
    $\tilde{\mathbf{n}}_N(0) = \mathbf{0}_N, \ldots, \tilde{\mathbf{n}}_N(0-L+1) = \mathbf{0}_N, d(0) = 0,$
    $\ldots, d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF},$
0)  $\mathbf{h}_N(0) = \mathbf{0}_N, E_N^{f(m)}(0) = d^2,$
    $\mathbf{h}_N^{f(m)}(0) = \mathbf{0}_N, \mathbf{h}_N^{b(m)}(0) = \mathbf{0}_N,$
    $m = 1, 2, \ldots, M, \mathbf{G}_{NF}^{(M)}(1) = \mathbf{0}_{NF}$
For  $k = 1, 2, \ldots, K$
For  $m = M, M-1, \ldots, 1$
1)  $\acute{\mathbf{a}}_F^{f(m)}(k) = \mathbf{x}_F^{(m)}(k) - \mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$
2)  $\acute{\mathbf{a}}_F^{b(m)}(k) = \mathbf{x}_F^{(m)}(k-N_m) - \mathbf{h}_N^{b(m)H}(k-1)\times$
    $\times\mathbf{X}_{NF}^{(m-1)}(k)$
3)  $\mathbf{h}_N^{f(m)}(k) = \mathbf{h}_N^{f(m)}(k-1) + \mathbf{G}_{NF}^{(m)}(k)\acute{\mathbf{a}}_F^{f(m)H}(k)$
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)  $E_N^{f(m)}(k) = lE_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k)\times$
    $\times\acute{\mathbf{a}}_F^{f(m)H}(k)$
6)  $\mathbf{G}_{(N+1)F}^{(m)}(k) = \left[1,\ -\mathbf{h}_N^{f(m)H}(k)\right]^H\times$
    $\times\mathbf{e}_F^{f(m)}(k)/E_N^{f(m)}(k) + \left[\mathbf{0}_F^T,\ \mathbf{G}_{NF}^{(m)H}(k)\right]^H$
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)  $\mathbf{G}_{NF}^{(m-1)}(k) = \left[\tilde{\mathbf{Q}}_{NF}^{(m)}(k) + \mathbf{h}_N^{b(m)}(k-1)\mathbf{q}_F^{(m)}(k)\right]\times$
    $\times\left[\mathbf{I}_F - \acute{\mathbf{a}}_F^{b(m)H}(k)\mathbf{q}_F^{(m)}(k)\right]^{-1}$
9)  $\mathbf{h}_N^{b(m)}(k) = \mathbf{h}_N^{b(m)}(k-1) + \mathbf{G}_{NF}^{(m-1)}(k)\acute{\mathbf{a}}_F^{b(m)H}(k)$
End for $m$
10) $\acute{\mathbf{a}}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$
11) $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + \mathbf{G}_{NF}^{(0)}(k)\acute{\mathbf{a}}_F^H(k)$
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 $\left[m^{0.5}x_m(k-L), x_m(k), m^{0.5}\boldsymbol{xr}_m(k-L), \boldsymbol{xr}_m(k)\right]$ and $\left[m^{0.5}x_m(k-N_m-L), x_m(k-N_m), m^{0.5}\boldsymbol{xr}_m(k-N_m-L), \boldsymbol{xr}_m(k-N_m)\right]$. The vectors $\mathbf{á}_F^{f(m)}(k)$, $\mathbf{e}_F^{f(m)}(k)$, $\mathbf{á}_F^{b(m)}(k)$, $\mathbf{q}_F^{f(m)}(k)$ and $\mathbf{q}_F^{(m)}(k)$ are row-vectors. The matrices $\mathbf{X}_{NF}^{(m)}(k)$ are composed as $\mathbf{X}_{NF}^{(m)}(k) = \left[m^{0.5}÷_N^{(m)}(k-L), ÷_N^{(m)}(k), m^{0.5}x\mathbf{ñ}_N^{(m)}(k-L), x\mathbf{ñ}_N^{(m)}(k)\right]$. The columns of the matrix are defined like the vectors $÷_N^{(m)}(k)$:

$$÷_N^{(0)}(k) = ÷_N(k),$$

$$÷_N^{(1)}(k) = \left[\mathbf{x}_{N_1}^T(k-1), \mathbf{x}_{N_2}^T(k), \ldots, \mathbf{x}_{N_m}^T(k), \ldots, \mathbf{x}_{N_M}^T(k)\right]^T,$$

$$\vdots$$

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

$$\vdots$$

$$÷_N^{(M)}(k) = \left[\mathbf{x}_{N_1}^T(k-1), \mathbf{x}_{N_2}^T(k-1), \ldots, \mathbf{x}_{N_m}^T(k-1), \ldots, \mathbf{x}_{N_M}^T(k-1)\right]^T.$$

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 [13].

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

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

$$\mathbf{Ö}_F^{(m-1)}(k) = \left[\mathbf{I}_F - l^{-1}\overline{\mathbf{Ö}}_F^{(m)}(k)\mathbf{q}_F^{(m)H}(k) \times \times \mathbf{á}_F^{b(m)}(k)\right]^{-1}\tilde{\mathbf{Ö}}_F^{(m)}(k).$$

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

$$\left[\overline{\mathbf{Ö}}_F^{(m)}(k)\right]^{-1} = \left[\mathbf{Ö}_F^{(m)}(k)\right]^{-1} + l^{-1}\mathbf{á}_F^{f(m)H}(k)\mathbf{ç}_F^{(m)}(k),$$

and

$$\left[\mathbf{Ö}_F^{(m-1)}(k)\right]^{-1} = \left[\overline{\mathbf{Ö}}_F^{(m)}(k)\right]^{-1} - l^{-1}\mathbf{á}_F^{b(m)H}(k)\mathbf{q}_F^{(m)}(k).$$

The matrices allow the calculations of Kalman gains

$$\mathbf{G}_{NF}(k) = \mathbf{G}_{NF}^{(0)}(k) = l^{-1}\tilde{\mathbf{T}}_{NF}^{(0)}(k)\mathbf{Ö}_F^{(0)}(k),$$

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

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

and

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

A multichannel parallel version of stabilized FAEST algorithm [14] is shown below.

```
0)  Init.: $÷_N(0) = \mathbf{0}_N, \ldots, ÷_N(0-L+1) = \mathbf{0}_N,$
    $\mathbf{ñ}_N(0) = \mathbf{0}_N, \ldots, \mathbf{ñ}_N(0-L+1) = \mathbf{0}_N, d(0) = 0,$
    $\ldots, d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF}, \mathbf{h}_N(0) = \mathbf{0}_N,$
    $E_N^{f(m)}(0) = d^2, E_N^{b(m)}(0) = d^2 l^{-N_m},$
    $\mathbf{h}_N^{f(m)}(0) = \mathbf{0}_N, \mathbf{h}_N^{b(m)}(0) = \mathbf{0}_N, m = 1,2,\ldots,M,$
    $\tilde{\mathbf{T}}_{NF}^{(M)}(1) = \mathbf{0}_{NF}, \mathbf{Ö}_F^{(M)}(1) = \mathbf{S}_F$
For $k = 1,2,\ldots,K$
For $m = M, M-1, \ldots, 1$
1)  $\mathbf{á}_F^{f(m)}(k) = \mathbf{x}_F^{(m)}(k) - \mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$
2)  $\mathbf{e}_F^{f(m)}(k) = \mathbf{á}_F^{f(m)}(k)\mathbf{Ö}_F^{(m)}(k)$
3)  $\mathbf{ç}_F^{(m)}(k) = \mathbf{á}_F^{f(m)}(k) / E_N^{f(m)}(k-1)$
4)  $\tilde{\mathbf{T}}_{(N+1)F}^{(m)}(k) = \begin{bmatrix} 1 \\ -\mathbf{h}_N^{f(m)}(k-1) \end{bmatrix}\mathbf{ç}_F^{(m)}(k) + \begin{bmatrix} \mathbf{0}_F^T \\ \tilde{\mathbf{T}}_{NF}^{(m)}(k) \end{bmatrix}$
5)  $\mathbf{S}_{N+1}^{(m)}\mathbf{T}_{N+1}^{(m)T}\left\{\tilde{\mathbf{T}}_{(N+1)F}^{(m)}(k)\right\} = \begin{bmatrix} \tilde{\mathbf{Q}}_{NF}^{(m)}(k) \\ \tilde{\mathbf{q}}_F^{(m)}(k) \end{bmatrix}$
6)  $\mathbf{h}_N^{f(m)}(k) = \mathbf{h}_N^{f(m)}(k-1) + l^{-1}\tilde{\mathbf{T}}_{NF}^{(m)}(k)\mathbf{e}_F^{f(m)H}(k)$
7)  $E_N^{f(m)}(k) = lE_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k)\mathbf{á}_F^{f(m)H}(k)$
8)  $\left[\overline{\mathbf{Ö}}_F^{(m)}(k)\right]^{-1} = \left[\mathbf{Ö}_F^{(m)}(k)\right]^{-1} + l^{-1}\mathbf{á}_F^{f(m)H}(k)\mathbf{ç}_F^{(m)}(k)$
9)  $\mathbf{á}_F^{b(m)}(k) = \mathbf{x}_F^{(m)}(k-N_m) - \mathbf{h}_N^{b(m)H}(k-1)\mathbf{X}_{NF}^{(m-1)}(k)$
10) $\mathbf{q}_F^{(m)}(k) = \mathbf{á}_F^{b(m)}(k) / E_N^{b(m)}(k-1)$
11) $\tilde{\mathbf{a}}_F^{b(m)}(k) = \tilde{\mathbf{q}}_F^{(m)}(k)E_N^{b(m)}(k-1)$
12) $\mathbf{á}_F^{b(1)(m)}(k) = K_1\mathbf{á}_F^{b(m)}(k) + (1-K_1)\tilde{\mathbf{a}}_F^{b(m)}(k)$
13) $\mathbf{á}_F^{b(2)(m)}(k) = K_2\mathbf{á}_F^{b(m)}(k) + (1-K_2)\tilde{\mathbf{a}}_F^{b(m)}(k)$
14) $\mathbf{á}_F^{b(5)(m)}(k) = K_5\mathbf{á}_F^{b(m)}(k) + (1-K_5)\tilde{\mathbf{a}}_F^{b(m)}(k)$
15) $\mathbf{t}_F^{(m)}(k) = K_4\mathbf{q}_F^{(m)}(k) + (1-K_4)\tilde{\mathbf{q}}_F^{(m)}(k)$
16) $\tilde{\mathbf{T}}_{NF}^{(m-1)}(k) = \tilde{\mathbf{Q}}_{NF}^{(m)}(k) + \mathbf{h}_N^{b(m)}(k-1)\mathbf{t}_F^{(m)}(k)$
17) $\left[\hat{\mathbf{Ö}}_F^{(m-1)}(k)\right]^{-1} = l^{-1}\tilde{\mathbf{T}}_{NF}^{(m-1)H}(k)\mathbf{X}_{NF}^{(m-1)}(k) + \mathbf{S}_F$
18) $\left[\tilde{\mathbf{Ö}}_F^{(m-1)}(k)\right]^{-1} = \left[\overline{\mathbf{Ö}}_F^{(m)}(k)\right]^{-1} -$
    $- l^{-1}\mathbf{á}_F^{b(5)(m)H}(k)\tilde{\mathbf{q}}_F^{(m)}(k)$
19) $\left[\mathbf{Ö}_F^{(m-1)}(k)\right]^{-1} = K_3\left[\hat{\mathbf{Ö}}_F^{(m-1)}(k)\right]^{-1} +$
    $+ (1-K_3)\left[\tilde{\mathbf{Ö}}_F^{(m-1)}(k)\right]^{-1}$
20) $\mathbf{e}_F^{b(1)(m)}(k) = \mathbf{á}_F^{b(1)(m)}(k)\tilde{\mathbf{Ö}}_F^{(m-1)}(k)$
21) $\mathbf{e}_F^{b(2)(m)}(k) = \mathbf{á}_F^{b(2)(m)}(k)\tilde{\mathbf{Ö}}_F^{(m-1)}(k)$
```

```
22) $E_N^{b(m)}(k) = lE_N^{b(m)}(k-1) + \mathbf{e}_F^{b(2)(m)}(k)\mathbf{á}_F^{b(2)(m)H}(k)$
23) $\mathbf{h}_N^{b(m)}(k) = \mathbf{h}_N^{b(m)}(k-1) + l^{-1}\tilde{\mathbf{T}}_{NF}^{(m-1)}(k)\mathbf{e}_F^{b(1)(m)H}(k)$
End for $m$
24) $\mathbf{á}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$
25) $\mathbf{e}_F(k) = \mathbf{á}_F(k)\mathbf{Ö}_F^{(0)}(k)$
25) $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + l^{-1}\tilde{\mathbf{T}}_{NF}^{(0)}(k)\mathbf{e}_F^H(k)$
25) $\tilde{\mathbf{T}}_{NF}^{(M)}(k+1) = \tilde{\mathbf{T}}_{NF}^{(0)}(k), \mathbf{Ö}_F^{(M)}(k+1) = \mathbf{Ö}_F^{(0)}(k)$
End for $k$
```

Using the duality between the fast RLS algorithms and the fast QRD based least squares algorithms [15], parallel multichannel version of the square root free inverse QRD fast RLS algorithm [16, 17] can be developed. The algorithm is presented below.

```
0)  Init.: $÷_N(0) = \mathbf{0}_N, \ldots, ÷_N(0-L+1) = \mathbf{0}_N,$
    $\mathbf{ñ}_N(0) = \mathbf{0}_N, \ldots, \mathbf{ñ}_N(0-L+1) = \mathbf{0}_N, d(0) = 0, \ldots,$
    $d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF}, \mathbf{h}_N(0) = \mathbf{0}_N,$
    $E_N^{f(m)}(0) = d^2, E_N^{b(m)}(0) = d^2 l^{-N_m}, \mathbf{h}_N^{f(m)}(0) = \mathbf{0}_N,$
    $\mathbf{h}_N^{b(m)}(0) = \mathbf{0}_N, m = 1,2,\ldots,M, \mathbf{G}_{NF}^{(M)}(1) = \mathbf{0}_{NF},$
    $\mathbf{K}_F^{B(M)}(1) = \mathbf{S}_F$
For $k = 1,2,\ldots,K$
For $m = M, M-1, \ldots, 1$
1) $\mathbf{á}_F^{f(m)}(k) = \mathbf{x}_F^{(m)}(k) - \mathbf{h}_N^{f(m)H}(k-1)\mathbf{X}_{NF}^{(m)}(k)$
2) $\mathbf{e}_F^{f(m)}(k) = \mathbf{á}_F^{f(m)}(k)\left[\mathbf{K}_F^{B(m)}(k)\right]^{-1}$
3) $\overline{\mathbf{K}}_F^{B(m)}(k) = \mathbf{K}_F^{B(m)}(k) + l^{-1}E_N^{-1f(m)}(k-1)\times$
   $\times\mathbf{á}_F^{f(m)H}(k)\mathbf{á}_F^{f(m)}(k)$
4) $\mathbf{C}_F^{f(m)}(k) = \mathbf{K}_F^{B(m)}(k)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}$
5) $\mathbf{s}_F^{f(m)}(k) = l^{-1}E_N^{-1f(m)}(k-1)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}\mathbf{á}_F^{f(m)H}(k)$
6) $\mathbf{Q}_{NF}^{f(m)}(k) = \mathbf{G}_{NF}^{(m)}(k)\mathbf{C}_F^{f(m)}(k) -$          or 8)
   $-\mathbf{h}_N^{f(m)}(k-1)\mathbf{s}_F^{f(m)H}(k)$
7) $\mathbf{h}_N^{f(m)}(k) = \mathbf{h}_N^{f(m)}(k-1) + \mathbf{G}_{NF}^{(m)}(k)\mathbf{á}_F^{f(m)H}(k)$
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) $E_N^{f(m)}(k) = lE_N^{f(m)}(k-1) + \mathbf{e}_F^{f(m)}(k)\mathbf{á}_F^{f(m)H}(k)$
12) $\mathbf{á}_F^{b(m)}(k) = \mathbf{x}_F^{(m)}(k-N_m) - \mathbf{h}_N^{b(m)H}(k-1)\mathbf{X}_{NF}^{(m-1)}(k)$
13) $\mathbf{K}_F^{B(m-1)}(k) = \overline{\mathbf{K}}_F^{B(m)}(k) - l^{-1}E_N^{-1b(m)}(k-1)\times$
    $\times\mathbf{á}_F^{b(m)H}(k)\mathbf{á}_F^{b(m)}(k)$
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) $\mathbf{s}_F^{b(m)}(k) = l^{-1}E_N^{-1b(m)}(k-1)\left[\overline{\mathbf{K}}_F^{B(m)}(k)\right]^{-1}\mathbf{á}_F^{b(m)H}(k)$
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) $\mathbf{e}_F^{b(m)}(k) = \mathbf{á}_F^{b(m)}(k)\left[\mathbf{K}_F^{B(m-1)}(k)\right]^{-1}$
18) $E_N^{b(m)}(k) = lE_N^{b(m)}(k-1) + \mathbf{e}_F^{b(m)}(k)\mathbf{á}_F^{b(m)H}(k)$
19) $\mathbf{h}_N^{b(m)}(k) = \mathbf{h}_N^{b(m)}(k-1) + \mathbf{G}_{NF}^{(m-1)}(k)\mathbf{á}_F^{b(m)H}(k)$
End for $m$
20) $\mathbf{á}_F(k) = \mathbf{d}_F(k) - \mathbf{h}_N^H(k-1)\mathbf{X}_{NF}(k)$
21) $\mathbf{h}_N(k) = \mathbf{h}_N(k-1) + \mathbf{G}_{NF}^{(0)}(k)\mathbf{á}_F^H(k)$
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 $k$
```

The identity of the parallel and the same named sequential fast RLS algorithms is ensured by means of the square matrices $\left[\mathbf{Ö}_F^{(m)}(k)\right]^{-1}$ 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 [10]. In the parallel LC RLS algorithms, MIL (6) is used for the calculation of the matrices $\mathbf{Ã}_{NJ}(k) = \mathbf{R}_N^{-1}(k)\mathbf{C}_{NJ}$, $\mathbf{Ø}_J^{-1}(k) = \left[\mathbf{C}_{NJ}^H\mathbf{Ã}_{NJ}(k)\right]^{-1}$ and $\mathbf{Q}_{NJ}(k) = \mathbf{Ã}_{NJ}(k)\mathbf{Ø}_J^{-1}(k)$, caused by linear constrains in (2).

The below LC RLS algorithm, based on the matrix $\mathbf{Q}_{NJ}(k)$ calculation, is also 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.: $÷_N(0) = \mathbf{0}_N, \ldots, ÷_N(0-L+1) = \mathbf{0}_N,$
    $\mathbf{ñ}_N(0) = \mathbf{0}_N, \ldots, \mathbf{ñ}_N(0-L+1) = \mathbf{0}_N,$
    $d(0) = 0, \ldots, d(0-L+1) = 0, \mathbf{X}_{NF}(0) = \mathbf{O}_{NF},$
    $\mathbf{R}_N^{-1}(0) = d^{-2}\mathbf{Ë}_N, \mathbf{Ã}_{NJ}(0) = \mathbf{R}_N^{-1}(0)\mathbf{C}_{NJ},$
    $\mathbf{Q}_{NJ}(0) = \mathbf{Ã}_{NJ}(0)\left[\mathbf{C}_{NJ}^H\mathbf{Ã}_{NJ}(0)\right]^{-1}, \mathbf{h}_N(0) = \mathbf{Q}_{NJ}(0)\mathbf{f}_J$
    $\mathbf{Ë}_N = diag(1, l, \ldots, l^{N_1-1}, \ldots, 1, l, \ldots, l^{N_M-1})$
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) $\mathbf{Q}'_{NJ}(k) = \left[\mathbf{Q}_{NJ}(k-1) - \mathbf{G}_{NF}(k)\mathbf{N}^{H}_{JF}(k)\right]\times$
   $\times\left[\mathbf{I}_{J} + \dfrac{\mathbf{V}_{JF}(k)\mathbf{N}^{H}_{JF}(k)}{\mathbf{I}_{F} - \mathbf{N}^{H}_{JF}(k)\mathbf{V}_{JF}(k)}\right]$
5) $\mathbf{Q}_{NJ}(k) = \mathbf{Q}'_{NJ}(k) + \mathbf{C}_{NJ}\left(\mathbf{C}^{H}_{NJ}\mathbf{C}_{NJ}\right)^{-1}\times$
   $\times\left[\mathbf{I}_{J} - \mathbf{C}^{H}_{NJ}\mathbf{Q}'_{NJ}(k)\right]$
6) $\acute{\mathbf{a}}_{F}(k) = \mathbf{d}_{F}(k) - \mathbf{h}^{H}_{N}(k-1)\mathbf{X}_{NF}(k)$
7) $\mathbf{h}'_{N}(k) = \mathbf{h}_{N}(k-1) + \mathbf{G}_{NF}(k)\acute{\mathbf{a}}^{H}_{F}(k)$
8) $\mathbf{h}_{N}(k) = \mathbf{h}'_{N}(k) + \mathbf{Q}_{NJ}(k)\left[\mathbf{f}_{J} - \mathbf{C}^{H}_{NJ}\mathbf{h}'_{N}(k)\right]$
End for $k$
```

4 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. Due to the identity, the simulations results, which confirm the presented algorithms efficiency, coincide with those of [3]. 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] A.H. Sayed, *Fundamentals of adaptive filtering*, Hoboken, NJ: John Wiley and Sons, Inc, 2003.

[2] J. Benesty, Y. Huang (Eds.), *Adaptive signal processing: applications to real-world problems*, New York: Springer-Verlag, 2003.

[3] V.I. Djigan, Multichannel RLS and fast RLS adaptive filtering algorithms, *Successes of Modern Radioelectronics*, No.11, 2004, pp. 48-77.

[4] Ya.Ya. Petrichkovich, T.V. Solokhina, MULTICORE signal controllers - new Russian family of SoC, *Proceedings of 6-th International Conference on Digital Signal Processing and its Applications*, Moscow, Russia, vol.1, 2004, pp. 8-15.

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

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

[7] V.I. Djigan, Multichannel fast RLS adaptive filtering algorithm for parallel implementation by means of four processors, *Proceedings of Bauman's Moscow State Technical University*, No.1, 2005, pp. 83-98.

[8] V.I. Djigan, Parallel regularized multichannel RLS adaptive filtering algorithms, *Digital Signal Processing*, No.2, 2004, pp. 7-13.

[9] V.I. Djigan, Parallelizable sliding window regularized fast RLS algorithm for multichannel linearly constrained adaptive filtering, *Proceedings of the 10-th International Conference on Radars, Navigation and Communication*, Voronezh, Russia, vol. 1, 2004, pp. 132-142.

[10] L.S. Resende, J.M.T. Romano, M.G. Bellanger, A fast least-squares algorithm for linearly constrained adaptive filtering, *IEEE Trans. Signal Processing*, vol.44, No.5, 1996, pp. 1168-1174.

[11] A.A. Giordano, F.M. Hsu, *Least square estimation with application to digital signal processing*, Toronto: John Wiley and Sons, Inc., 1985.

[12] G. Zelniker, F.J.Taylor, *Advanced digital signal processing: theory and applications*, New York: Marcel Dekker, Inc., 1994.

[13] G.-O.A. Glentis., N. Kalouptsidis, Fast adaptive algorithms for multichannel filtering and system identification, *IEEE Trans. Signal Processing*, vol.40, No.10, 1992, pp. 2433-2458.

[14] D.T.M. Slock, T. Kailath, Numerically stable fast transversal filters for recursive least squares adaptive filtering, *IEEE Trans. Signal Processing*, vol.39, No.1, 1991, pp. 92-114.

[15] G.-O.A. Glentis, On the duality between the fast transversal and the fast QRD adaptive least squares algorithms, *IEEE Trans. Signal Processing*, vol. 47, No.8, 1999, pp. 2317-2321.

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

[17] S.F. Hsieh, K.J.R. Liu, A unified square-root-free approach for QRD based recursive least squares estimation, *IEEE Trans. Signal Processing*, vol.41, No.3, 1993, pp. 1405-1409.