# The Fast Subsampled-Updating Fast Transversal Filter (FSU FTF) RLS Algorithm[†]

Dirk T.M. Slock and Karim Maouche

Institut Eurecom, B.P. 193, 06904 Sophia Antipolis Cedex, France.

Please send all correspondence to:

Dirk Slock

Institut Eurecom
2229 route des Crêtes
B.P. 193
F-06904 Sophia Antipolis Cedex
France

slock@eurecom.fr

[†]This work was supported in part by the CNET, France, under contract nr 93 6 B 005. This work has also benefitted from cooperations within the group of signal processing researchers GDR 134 TdSI (part of the french CNRS).

French title: Le FSU FTF: un Algorithme des Moindres Carrés Récursifs Doublement Rapide

Keywords: Recursive least-squares, Fast transversal filter, Schur algorithm, Fast Fourier transform, Block processing, Acoustic echo cancellation.

Mots Clés: Moindres carrés récursifs, Algorithmes rapides, Algorithme de Schur, Transformée de Fourier rapide, Traitement en bloc, Annulation d'écho acoustique

# The Fast Subsampled-Updating Fast Transversal Filter (FSU FTF) RLS Algorithm

Dirk T.M. Slock and Karim Maouche

## Abstract

We present a new fast algorithm for Recursive Least-Squares (RLS) adaptive filtering that uses displacement structure and subsampled updating. The FSU FTF algorithm is based on the Fast Transversal Filter (FTF) algorithm, which exploits the shift invariance that is present in the RLS adaptation of a FIR filter. The FTF algorithm is in essence the application of a rotation matrix to a set of filters and in that respect resembles the Levinson algorithm. In the subsampled updating approach, we accumulate the rotation matrices over some time interval before applying them to the filters. It turns out that the successive rotation matrices themselves can be obtained from a Schur type algorithm which, once properly initialized, does not require inner products. The various convolutions that thus appear in the algorithm are done using the Fast Fourier Transform (FFT). For relatively long filters, the computational complexity of the new algorithm is smaller than the one of the well-known LMS algorithm, rendering it especially suitable for applications such as acoustic echo cancellation.

## Résumé

Nous présentons un nouvel algorithme rapide des moindres carrés récursifs (MCR), basé sur la structure de déplacement et la mise à jour sous-échantillonnée. Le FSU FTF est dérivé à partir de l'algorithme FTF qui exploite une certaine propriété d'invariance sous l'opération de décalage qui est inhérente au filtrage adaptatif par les MCR. L'algorithme FTF peut être vu comme l'application d'une matrice de rotation à un ensemble de filtres et s'apparente de ce point de vue à l'algorithme de Levinson. Dans une approche de mise à jour du filtre adaptatif sous-échantillonnée, nous accumulons pendant un bloc d'échantillons les matrices de rotation successives puis, nous appliquons la matrice résultat aux filtres. Ces matrices de rotation peuvent être obtenues en utilisant une procédure de type Schur. On évite ainsi les calculs de produits scalaires, sauf pour l'initialisation de cet algorithme FTF-Schur. Les convolutions qui apparaissent ainsi à différents endroits dans l'algorithme sont effectuées à l'aide de la Transformée de Fourier Rapide (TFR). Pour des filtres relativement longs, la complexité du nouvel algorithme est plus faible que celle du LMS, ce qui le rend très adapté pour la résolution de problèmes tels que celui de l'annulation d'écho acoustique.

## 1 Introduction

Adaptive filtering algorithms consist of a joint-process or filtering part, in which first of all an error signal is computed as the difference between the desired-response signal and the output of the adaptive filter, computed with the latest estimate of the filter coefficients. Then this error signal is multiplied with a gain vector to form the gradient in the filter update equation. In the RLS algorithm [2], the gain vector is the product of the Kalman gain and a likelihood variable. In the conventional RLS algorithm, these quantities are computed from the sample covariance matrix, the inverse of which is updated via the Riccati equation. This requires $\mathcal{O}(N^2)$ computations for a FIR filter length equal to $N$. Fast RLS algorithms such as the Fast Transversal Filter (FTF) algorithm [1] exploit a certain shift invariance structure in the input data vector, which is inherited by the sample input covariance and crosscorrelation matrices. Using this shift-invariance, the gain update part of RLS becomes a prediction part (involving forward and backward prediction problems) in FTF, the complexity of which is reduced from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$. Other fast RLS algorithms such as the Fast Lattice (FLA) and Fast QR (FQR) algorithms [2],[5] also provide the same filtering error signal, but replace the transversal filter coefficients by a transformed set of parameters as in the square-root Kalman filtering/RLS algorithms.

In [10],[8], we have pursued an alternative way to reduce the complexity of RLS adaptive filtering algorithms. The approach consists of subsampling the filter adaptation, i.e. the LS filter estimate is no longer provided every sample but every $L \geq 1$ samples (subsampling factor $L$). This leads to the Subsampled-Updating RLS (SU RLS) algorithm, which nevertheless provides exactly the same filtering error signal as the RLS algorithm. The computational complexity of the SU RLS algorithm is certainly not reduced w.r.t. to that of the RLS algorithm. However, in the SU RLS algorithm the Kalman gain and the likelihood variable are $L \times N$ and $L \times L$ matrices resp. which, due to the shift invariance present in the problem, exhibit a low displacement rank. Hence, by using the displacement structure and the FFT (when computing convolutions), we have derived a fast version of SU RLS that we have called the FSU RLS algorithm.

Here, we propose a dual strategy, see Fig. 1. Namely, after having exploited shift-invariance in the RLS algorithm to obtain the FTF algorithm, we shall apply subsampled updating to the estimation of the filters involved. The starting point is an interpretation of the FTF algorithm as a rotation applied to the vectors of filter coefficients. Using the filter estimates at a certain time instant, we compute the filter outputs over the next $L$ time instants. Using what we shall call a FTF-Schur algorithm, it will be possible to compute from these multi-step ahead predicted filter outputs the one step ahead predicted filter outputs, without updating or using the filters. These quantities will allow us to compute the successive rotation matrices of the FTF algorithm for the next $L$ time instants. Because of the presence of a shift operation in the FTF algorithm, it turns out to be most convenient to work with the $z$-transform of the rotation matrices and the filters. One rotation matrix is then a polynomial matrix of order one, and the product of $L$ successive rotation matrices is a polynomial matrix of order $L$. Applying the $L$ rotation matrices to the filter vectors becomes an issue of multiplying polynomials, which can be efficiently carried out using the FFT.

The subsampled updating technique turns out to be especially applicable in the case of very long filters such as occur in the acoustic echo cancellation problem. The computational gain it offers is obtained in exchange for some processing delay, as is typical of block processing. In order to formulate the RLS adaptive filtering problem and to fix notation, we shall first recall the RLS algorithm.

## 2 The RLS Algorithm

An adaptive transversal filter $W_{N,k}$ forms a linear combination of $N$ consecutive input samples $\{x(i-n), n = 0, \ldots, N-1\}$ to approximate (the negative of) the desired-response signal $d(i)$. The resulting error signal is given by

$$\epsilon_N(i|k) \;=\; d(i) + W_{N,k}\, X_N(i) \;=\; d(i) + \sum_{n=0}^{N-1} W_{N,k}^{n+1}\, x(i-n) \tag{1}$$

where $X_N(i) = \left[x^H(i)\; x^H(i-1) \cdots x^H(i-N+1)\right]^H$ is the input data vector and superscript $^H$ denotes Hermitian (complex conjugate) transpose. In the RLS algorithm, the set of $N$

transversal filter coefficients $W_{N,k} = \left[W^1_{N,k} \cdots W^N_{N,k}\right]$ are adapted so as to minimize recursively the following LS criterion

$$\begin{aligned} \xi_N(k) &= \min_{W_N} \left\{ \sum_{i=1}^{k} \lambda^{k-i} \left\| d(i) + W_N\, X_N(i) \right\|^2 \; + \; \lambda^{k+1}\mu \left\| W_N - W_0 \right\|^2_{\Lambda_N} \right\} \\ &= \sum_{i=1}^{k} \lambda^{k-i} \left\| \epsilon_N(i|k) \right\|^2 \; + \; \lambda^{k+1}\mu \left\| W_{N,k} - W_0 \right\|^2_{\Lambda_N} \end{aligned} \tag{2}$$

where $\lambda \in (0,1]$ is the exponential weighting factor, $\mu > 0$, $\Lambda_N = \operatorname{diag}\left\{\lambda^{N-1}, \ldots, \lambda, 1\right\}$, $\|v\|^2_\Lambda = v\Lambda v^H$, $\|.\| = \|.\|_I$. The second term in the LS criterion represents a priori information. For instance, prior to measuring the signals, we may assume that $W_N$ is distributed as $W_N \sim \mathcal{N}\left(W_0, R_0^{-1}\right)$, $R_0 = \mu\lambda\Lambda_N$. The particular choice for $R_0$ will become clear in the discussion of the initialization of the FSU FTF algorithm. Minimization of the LS criterion leads to the following minimizer

$$W_{N,k} \;=\; -P^H_{N,k} R^{-1}_{N,k} \tag{3}$$

where

$$\begin{aligned} R_{N,k} &= \sum_{i=1}^{k} \lambda^{k-i} X_N(i) X_N^H(i) \; + \lambda^{k+1}\mu\Lambda_N \\ &= \lambda R_{N,k-1} + X_N(k) X_N^H(k)\,, \qquad\qquad R_{N,0} = R_0 = \mu\lambda\Lambda_N \\ P_{N,k} &= \sum_{i=1}^{k} \lambda^{k-i} X_N(i) d^H(i) \; - \lambda^{k+1}\mu\Lambda_N W_0^H \\ &= \lambda P_{N,k-1} + X_N(k) d^H(k)\,, \qquad\qquad P_{N,0} = -R_0 W_0^H \end{aligned} \tag{4}$$

are the sample second order statistics. Substituting the time recursions for $R_{N,k}$ and $P_{N,k}$ from (4) into (3) and using the matrix inversion lemma [3, pg 656] for $R^{-1}_{N,k}$, we obtain the RLS algorithm:

$$\widetilde{C}_{N,k} \;=\; -X_N^H(k)\lambda^{-1} R^{-1}_{N,k-1} \tag{5}$$

$$\gamma_N^{-1}(k) \;=\; 1 - \widetilde{C}_{N,k} X_N(k) \tag{6}$$

$$R^{-1}_{N,k} \;=\; \lambda^{-1} R^{-1}_{N,k-1} \;-\; \widetilde{C}^H_{N,k} \gamma_N(k) \widetilde{C}_{N,k} \tag{7}$$

$$\epsilon^p_N(k) \;=\; \epsilon_N(k|k-1) \;=\; d(k) + W_{N,k-1} X_N(k) \tag{8}$$

$$\epsilon_N(k) \;=\; \epsilon_N(k|k) \;=\; \epsilon^p_N(k)\,\gamma_N(k) \tag{9}$$

$$W_{N,k} \;=\; W_{N,k-1} + \epsilon_N(k) \widetilde{C}_{N,k} \tag{10}$$

where $\epsilon^p_N(k)$ and $\epsilon_N(k)$ are the a priori and a posteriori error signals (resp. predicted and filtered errors in the Kalman filtering terminology) and one can verify (or see [1]) that they are related by the likelihood variable $\gamma_N(k)$ as in (9).

## 3 The Fast Transversal Filter Algorithm

The FTF algorithm efficiently exploits the shift-invariance present in the adaptive FIR filtering problem, which translates into a low displacement rank of $R^{-1}_{N,k}$. The computational complexity of the FTF algorithm is $7N$ in its most efficient form. The developments in this paper are immediately extendible to the multichannel case starting from the multichannel FTF algorithm [1],[7]. However, we shall restrict the formulas to the single-channel case for notational simplicity. The algorithm can be described in the following way, which emphasizes its rotational structure:

$$\begin{bmatrix} \left[\widetilde{C}_{N,k}\ 0\right] \\ A_{N,k} \\ B_{N,k} \\ \left[W_{N,k}\ 0\right] \end{bmatrix} = \Theta_k \begin{bmatrix} \left[0\ \widetilde{C}_{N,k-1}\right] \\ A_{N,k-1} \\ B_{N,k-1} \\ \left[W_{N,k-1}\ 0\right] \end{bmatrix}$$

$$\begin{aligned} \alpha_N(k) &= \lambda\alpha_N(k-1) + e_N(k)\, e^{pH}_N(k) \\ \beta_N(k) &= \lambda\beta_N(k-1) + r_N(k)\, r^{pH}_N(k) \\ \gamma_N(k) &= \lambda^N \beta_N(k) / \alpha_N(k) \end{aligned} \tag{11}$$

where $A_{N,k}$ and $B_{N,k}$ are the forward and backward prediction filters, $e^p_N(k)$ ($e_N(k)$) and $r^p_N(k)$ ($r_N(k)$) are the a priori (a posteriori) forward and backward predition errors and $\alpha_N(k)$ and $\beta_N(k)$ the corresponding prediction error variances. $\Theta_k$ is a $4 \times 4$ rotation matrix given by

$$\Theta_k = \Theta^4_k\, \Theta^3_k\, \Theta^2_k\, \Theta^1_k \tag{12}$$

where the four $4 \times 4$ matrices $\Theta_k^i \;\; i = 1,2,3,4$ are

$$\Theta_k^4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \epsilon_N(k) & 0 & 0 & 1 \end{bmatrix} \qquad \Theta_k^3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ r_N(k) & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Theta_k^2 = \begin{bmatrix} 1 & 0 & \frac{r_N^p(k)}{\lambda\beta_N(k-1)} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \Theta_k^1 = \begin{bmatrix} 1 & -\frac{e_N^p(k)}{\lambda\alpha_N(k-1)} & 0 & 0 \\ e_N(k) & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{13}$$

The likelihood variable $\gamma_N(k)$ can also be computed in two other ways [7] and we shall use one of these further on. In order to compute the rotation matrices, one must obtain the a priori errors $e_N^p(k), r_N^p(k)$ and $\epsilon_N^p(k)$ which are the outputs at time $k$ of the filters $A_{N,k-1}, B_{N,k-1}$ and $W_{N,k-1}$.

## 4 The FTF-Schur Algorithm

Now we introduce subsampled updating and from the filters at time instant $k-L$, we want to obtain the filters at time instant $k$. This will require the rotation matrices and hence the a priori errors in that time range. We shall show that these quantities can be computed without generating the intermediate filter estimates using a FTF-Schur algorithm.

Let us introduce the negative of the filter output

$$\widehat{d}_N^p(k) = d(k) - \epsilon_N^p(k) \tag{14}$$

$$\widehat{d}_N(k) = d(k) - \epsilon_N(k) \quad . \tag{15}$$

Consider now the following set of filtering operations

$$F_L(k) \triangleq \begin{bmatrix} \eta_{N,L,k}^H \\ e_{N,L,k}^{p\,H} \\ r_{N,L,k}^{p\,H} \\ -\widehat{d}_{N,L,k}^{p\,H} \end{bmatrix} \triangleq \begin{bmatrix} \left[0 \; \widetilde{C}_{N,k-L}\right] \\ A_{N,k-L} \\ B_{N,k-L} \\ \left[W_{N,k-L} \; 0\right] \end{bmatrix} X_{N+1,L,k}^H \tag{16}$$

where

$$X_{N+1,L,k} \;=\; \left[X_{N+1}(k-L+1)\cdots X_{N+1}(k)\right]^{H} \tag{17}$$

is the $L \times (N+1)$ Toeplitz input data matrix. $F_L(k)$ is a $4 \times L$ matrix the rows of which are the result of the filtering of the data sequence $\{x(j)\;,\; j = k-N-L+1, \cdots, k\}$ by the four filters of the FTF algorithm as indicated in Fig. 2. $\eta_{N,L,k}$ is the output of the Kalman gain and $e^p_{N,L,k}$ and $r^p_{N,L,k}$ are respectively the vectors of forward and backward prediction errors

$$e^p_{N,L,k} \;=\; \begin{bmatrix} e^H_N(k-L+1|k-L) \\ \vdots \\ e^H_N(k|k-L) \end{bmatrix} \;,\quad r^p_{N,L,k} \;=\; \begin{bmatrix} r^H_N(k-L+1|k-L) \\ \vdots \\ r^H_N(k|k-L) \end{bmatrix} \;. \tag{18}$$

The last row of $F_L(k)$ corresponds to the (multi-step ahead predicted) adaptive filter outputs

$$-\,\widehat{d}^{\,p}_{N,L,k} \;=\; \epsilon^p_{N,L,k} \;-\; d_{L,k} \;=\; \begin{bmatrix} -\widehat{d}^H_N(k-L+1|k-L) \\ \vdots \\ -\widehat{d}^H_N(k|k-L) \end{bmatrix} \;. \tag{19}$$

The first column of $F_L\,(k)$ is

$$F_L\,(k)\;\; u_{L,1} \;=\; \begin{bmatrix} 1-\gamma_N^{-1}\,(k-L) \\ e^p_N\,(k-L+1) \\ r^p_N\,(k-L+1) \\ -\widehat{d}^{\,p}_N\,(k-L+1) \end{bmatrix} \tag{20}$$

where $u_{L,n}$ is the $L\times 1$ vector with 1 at the $n^{th}$ position and 0 elsewhere. So with the quantities in $F_L\,(k)\;\; u_{L,1}$ and using the recursions for $\alpha_N$, $\beta_N$ and $\gamma_N^{-1}$, viz.

$$\alpha_N\,(k-L+1) \;=\; \lambda\alpha_N\,(k-L) + e^p_N\,(k-L+1)\,\gamma_N\,(k-L)\,e^{pH}_N\,(k-L+1) \tag{21}$$

$$\beta_N\,(k-L+1) \;=\; \lambda\beta_N\,(k-L) + r^p_N\,(k-L+1)\,\gamma_N\,(k-L+1)\,r^{pH}_N\,(k-L+1) \tag{22}$$

$$\gamma_N^{-1}(k-L+1) \;=\; \gamma_N^{-1}(k-L) + \frac{e^{pH}_N(k-L+1)\;e^p_N(k-L+1)}{\lambda\alpha_N(k-L)} - \frac{r^{pH}_N(k-L+1)\;r^p_N(k-L+1)}{\lambda\beta_N(k-L)} \tag{23}$$

it is possible to construct $\Theta_{k-L+1}$. Now we rotate both expressions for $F_L(k)$ in (16) with

$\Theta_{k-L+1}$ to obtain $\Theta_{k-L+1}F_L(k)$ which equals

$$\begin{bmatrix} \left[\widetilde{C}_{N,k-L+1}\ 0\right] \\ \\ A_{N,k-L+1} \\ \\ B_{N,k-L+1} \\ \\ \left[W_{N,k-L+1}\ 0\right] \end{bmatrix} X^H_{N+1,L,k} = \begin{bmatrix} \boxed{\eta^H_{N,L-1,k}} & * \\ e_N\,(k-L+1) & \boxed{e^{p\,H}_{N,L-1,k}} \\ r_N\,(k-L+1) & \boxed{r^{p\,H}_{N,L-1,k}} \\ -\widehat{d}_N\,(k-L+1) & \boxed{-\widehat{d}^{\,p\,H}_{N,L-1,k}} \end{bmatrix} . \tag{24}$$

Or we can write more compactly

$$\mathcal{S}\left(\Theta_{k-L+1}\,F_L(k)\right) \;=\; F_{L-1}(k) \tag{25}$$

where the operator $\mathcal{S}(M)$ stands for: shift the first row of the matrix $M$ one position to the right and drop the first column of the matrix thus obtained. Now this process can be repeated until we get $F_0(k)$ which is a matrix with no dimensions. So the same rotations that apply to the filters at times $k-l$, $l = L-1, \ldots, 0$, also apply to the set of filtering error vectors $F_l(k)$ over the same time span. Furthermore, at each rotation instance, the rotation parameters can be calculated from the first column of $F_l(k)$ and some auxiliary scalars ($\alpha$, $\beta$ and $\gamma$). Inner products (filtering operations) are only needed for the initialization (computation of $F_L(k)$). This is the FTF-Schur algorithm, which contrasts with the Levinson-style FTF algorithm in (11).

## 5 The FSU FTF Algorithm

Once we have computed the $L$ consecutive rotation matrices with the FTF-Schur algorithm, we want to apply them all at once to obtain the filters at time $k$ from the filters at time $k-L$. Due to the shift of the Kalman gain in (11), we need to work in the $z$-transform domain. So we shall associate polynomials with the filter coefficients as follows

$$\begin{bmatrix} \widetilde{C}_k(z) \\ A_k(z) \\ B_k(z) \\ W_k(z) \end{bmatrix} = \begin{bmatrix} \left[\widetilde{C}_{N,k}\ 0\right] \\ A_{N,k} \\ B_{N,k} \\ \left[W_{N,k}\ 0\right] \end{bmatrix} \begin{bmatrix} 1 \\ z^{-1} \\ \vdots \\ z^{-N} \end{bmatrix} . \tag{26}$$

Hence (11) can be written in the $z$-transform domain as

$$\begin{bmatrix} \widetilde{C}_k(z) \\ A_k(z) \\ B_k(k) \\ W_k(z) \end{bmatrix} = \Theta_k \begin{bmatrix} z^{-1} & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} \widetilde{C}_{k-1}(z) \\ A_{k-1}(z) \\ B_{k-1}(z) \\ W_{N,k-1}(z) \end{bmatrix} . \tag{27}$$

It appears natural to introduce

$$\Theta_k(z) = \Theta_k \begin{bmatrix} z^{-1} & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} . \tag{28}$$

Now, in order to adapt the filters at time $k$ from the ones at time $k-L$, we get straightforwardly

$$\begin{bmatrix} \widetilde{C}_k(z) \\ A_k(z) \\ B_k(k) \\ W_k(z) \end{bmatrix} = \Theta_{k,L}(z) \begin{bmatrix} \widetilde{C}_{k-L}(z) \\ A_{k-L}(z) \\ B_{k-L}(z) \\ W_{N,k-L}(z) \end{bmatrix} \tag{29}$$

where

$$\Theta_{k,L}(z) = \Theta_k(z)\,\Theta_{k-1}(z)\cdots\Theta_{k-L+1}(z) \quad . \tag{30}$$

As mentioned before, the successive rotation matrices can be obtained via the FTF-Schur algorithm with a computational complexity of $2.5L^2$ operations (counting only the most significant term as we often do), which takes into account the fact that a rotation matrix in factored form as in (13) only contains five non-trivial entries. Now also remark that $\Theta_{k,L}(z)$

has the following structure

$$\Theta_{k,L}\left(z\right) = \begin{bmatrix} * & * & * & 0 \\ * & * & * & 0 \\ * & * & * & 0 \\ * & * & * & 1 \end{bmatrix} \tag{31}$$

where the stars stand for polynomials in $z^{-1}$ of degree at most $L$. Taking into account these two remarks, the accumulation of the successive rotation matrices to form $\Theta_{k,L}\left(z\right)$ takes $7.5L^2$ operations.

As a result of the structure displayed in (31), the product in (29) represents 12 convolutions of a polynomial of order $L$ with a polynomial of order $N$. These convolutions can be done using fast convolution techniques [4]. In the case we consider, in which the orders of the polynomials are relatively large, we will implement the convolutions using the FFT technique. In that case the complexity for the update of each one of the four filters is $3(1+2\frac{N+1}{L})FFT\left(2L\right)+2\left(N+1\right))$ (multiply/add) operations plus $6\left(N+1\right)$ additions ($FFT(m)$ denotes the computational complexity for computing a FFT of length $m$, and we assume that $L$ is a power of 2 and that $\frac{N+1}{L}$ is an integer). The computation of $F_L(k)$ in (16) can also be done with the FFT and one should compute the FFTs of the filters only once. In the Overlap-Save method, the data matrix is decomposed into $\frac{N+1}{L}$ blocks of $L \times L$ Toeplitz matrices, which are then embedded into $2L \times 2L$ Toeplitz matrices. Note that at time $k$, only the most recent $2L$ samples of the input signal, corresponding to the new $L \times L$ block in the data matrix, have to be Fourier transformed. The other parts have been computed at previous instants (see [8] for more details). The resulting FSU FTF algorithm is summarized in Table I.

The initialization of the algorithm is done with the soft constraint initialization technique [1]. The additon of the soft constraint to the LS cost function as shown in (2) can be interpreted as the result of an unconstrained LS problem where the input signal is equal to $\sqrt{\mu}$ at time $k = -N$ and zero at all other time instants before time $k = 0$. Hence the FSU FTF algorithm departs from the following initial conditions

$$\begin{aligned}
&W_{N,0} = W_0 \\
&A_{N,0} = [1\ 0 \cdots 0]\ ,\ \alpha_N(0) = \lambda^N \mu \\
&B_{N,0} = [0 \cdots 0\ 1]\ ,\ \beta_N(0) = \mu \\
&\widetilde{C}_{N,0} = [0 \cdots 0]\ ,\ \gamma_N(0) = 1
\end{aligned} \tag{32}$$

With this initialization at $k = 0$, the corresponding initial sample covariance matrix is indeed $R_0 = \mu\lambda\Lambda_N$.

## 6 Concluding Remarks

The complexity of the FSU FTF is $\mathcal{O}((8\frac{N+1}{L}+17)\frac{FFT(2L)}{L} + 32\frac{N}{L} + 10L)$ operations per sample. This can be very interesting for long filters. For example, when $(N, L) = (4095, 256)$, $(8191, 256)$ and the FFT is done via the split radix ($FFT(2m) = m log_2(2m)$ real multiplications for real signals) the multiplicative complexity is respectively $1.1N$ and $0.74N$ per sample. This should be compared to $7N$ for the FTF algorithm, the currently fastest RLS algorithm, and $2N$ for the LMS algorithm. The number of additions is somewhat higher. The cost we pay is a processing delay which is of the order of $L$ samples. We have simulated the algorithm and have verified that it works. Preliminary experience appears to indicate that the numerical behavior of the algorithm may require further attention. This is not a surprising fact since the accumulation of round-off errors in the original FTF algorithm is known to be unstable [7]. One way to overcome this drawback is to use a rescuing procedure (see [1],[6],[8]). A more elaborate investigation of the numerical issues is the subject of ongoing research. In [10],[9],[8], we have introduced the FSU RLS algorithm, an alternative algorithm with a very similar computational complexity, but a very different internal structure. These developments leads us to conjecture that perhaps a lower bound on computational complexity has been reached for RLS algorithms when the subsampled updating strategy is applied and when the filters to be adapted are relatively long.

## References

[1] J.M. Cioffi and T. Kailath. "Fast, recursive least squares transversal filters for adaptive filtering". *IEEE Trans. on ASSP*, ASSP-32(2):304–337, April 1984.

[2] S. Haykin. *Adaptive Filter Theory.* Prentice-Hall, Englewood Cliffs, NJ, 1991. second edition.

[3] T. Kailath. *Linear Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1980.

[4] A.V. Oppenheim and R.W. Schafer. *Digital Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1975.

[5] D.T.M. Slock. "Reconciling Fast RLS Lattice and QR Algorithms". In *Proc. ICASSP 90 Conf.*, pages 1591–1594, Albuquerque, NM, April 3–6 1990.

[6] D.T.M. Slock. "Backward Consistency Concept and Round-Off Error Propagation Dynamics in Recursive Least-Squares Algorithms". *Optical Engineering*, 31(6):1153–1169, June 1992.

[7] D.T.M. Slock and T. Kailath. "Numerically Stable Fast Transversal Filters for Recursive Least-Squares Adaptive Filtering". *IEEE Trans. Signal Proc.*, ASSP-39(1):92–114, Jan. 1991.

[8] D.T.M. Slock and K. Maouche. "The Fast Subsampled-Updating Recursive Least-Squares (FSU RLS) for Adaptive Filtering Based on Displacement Structure and the FFT". Submitted to Signal Processing.

[9] D.T.M. Slock and K. Maouche. "The Fast Subsampled-Updating Recursive Least-Squares (FSU RLS) and Fast Transversal Filter (FSU FTF) Algorithms for Adapting Long FIR Filters". In André Gilloire, editor, *Proc. Third International Workshop on Acoustic Echo Control*, pages 75–86, Plestin les Grèves, France, Sept. 7-8 1993.

[10] D.T.M. Slock and K. Maouche. “Un Algorithme des Moindres Carrés Récursif Rapide: le FSU RLS”. In *Proc. Quatorzième Colloque sur le Traitement du Signal et des Images*, pages 495–498, Juan-les-Pins, France, Sept. 13-16 1993.

| **Table I: the FSU FTF Algorithm** | | |
|---|---|---|
| **#** | **Computation** | **Cost per L samples** |
| 1 | $\begin{bmatrix} \eta^H_{N,L,k} \\ e^{p\,H}_{N,L,k} \\ r^{p\,H}_{N,L,k} \\ -\widehat{d}^{\,p\,H}_{N,L,k} \end{bmatrix} = \begin{bmatrix} \left[0\ \widetilde{C}_{N,k-L}\right] \\ A_{N,k-L} \\ B_{N,k-L} \\ \left[W_{N,k-L}\ 0\right] \end{bmatrix} X^H_{N+1,L,k}$ | $(5+4\frac{N+1}{L})FFT(2L)+8N$ |
| 2 | FTF-Schur Algorithm:<br>Input: $\eta_{N,L,k},\ e^p_{N,L,k},\ r^p_{N,L,k},\ -\widehat{d}^{\,p}_{N,L,k}$<br>Output: $\Theta_{k-i}(z)\ \ i=L{-}1,\cdots,0$ | $2.5L^2$ |
| 3 | $\Theta_{k,L}(z) = \prod_{i=0}^{L-1}\Theta_{k-i}(z)$ | $7.5L^2$ |
| 4 | $\begin{bmatrix} \widetilde{C}_k(z) \\ A_k(z) \\ B_k(z) \\ W_k(z) \end{bmatrix} = \Theta_{k,L}(z) \begin{bmatrix} \widetilde{C}_{k-L}(z) \\ A_{k-L}(z) \\ B_{k-L}(z) \\ W_{N,k-L}(z) \end{bmatrix}$ | $(12+4\frac{N+1}{L})FFT(2L)+24N$ |
| Total cost per sample | | $(17+8\frac{N+1}{L})\frac{FFT(2L)}{L}+32\frac{N}{L}+10L$ |

# List of Figures

$$\begin{tikzcd}
 & \text{RLS} \arrow[dl, "\begin{array}{c}\text{block processing}\\ \text{(FFT)}\end{array}"] \arrow[dr, "\begin{array}{c}\text{displacement}\\ \text{structure}\end{array}"] & \\
\text{SU RLS} \arrow[d, "\begin{array}{c}\text{displacement}\\ \text{structure}\end{array}"'] & & \text{FTF} \arrow[d, "\begin{array}{c}\text{block processing}\\ \text{(FFT)}\end{array}"] \\
\text{FSU RLS} & & \text{FSU FTF}
\end{tikzcd}$$

Figure 1:

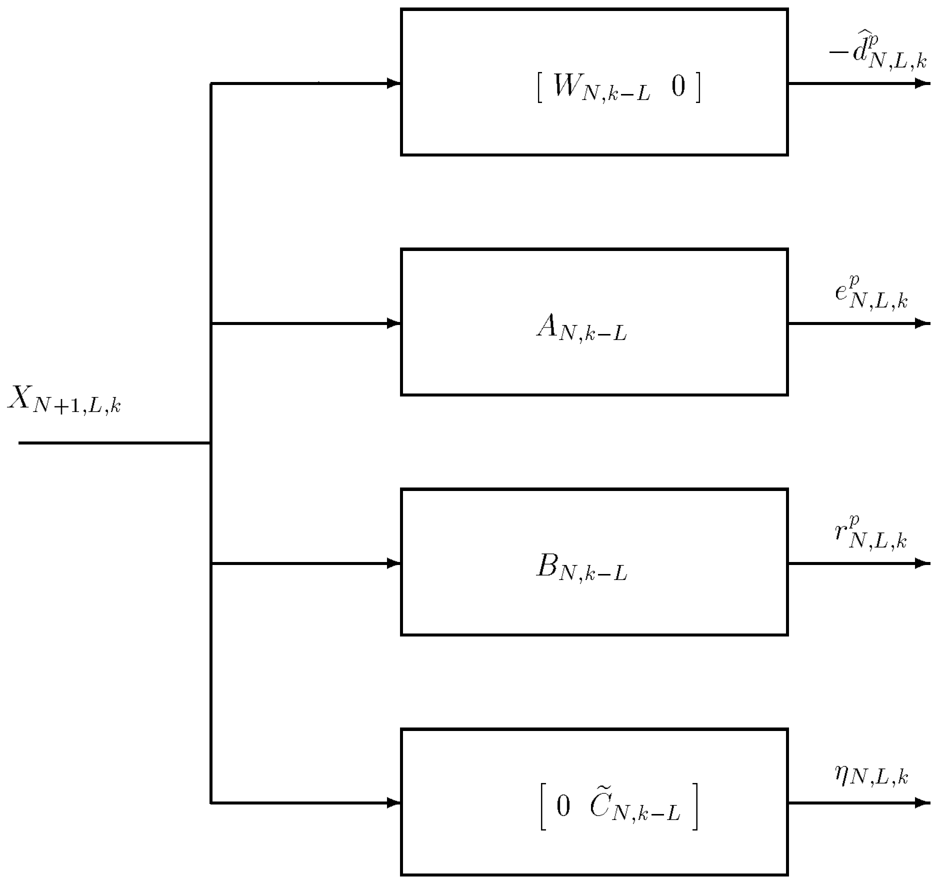
$X_{N+1,L,k}$
$[\, W_{N,k-L} \;\; 0 \,]$
$A_{N,k-L}$
$B_{N,k-L}$
$\left[\, 0 \;\; \widetilde{C}_{N,k-L} \,\right]$
$-\widehat{d}^{p}_{N,L,k}$
$e^{p}_{N,L,k}$
$r^{p}_{N,L,k}$
$\eta_{N,L,k}$

Figure 2:

# Biographies

**Dirk T.M. Slock** received the degree of Engineer in Electrical Engineering from the University of Gent, Belgium, in 1982, the M.S. degrees in Electrical Engineering and Statistics and the Ph.D. degree in Electrical Engineering from Stanford University, Stanford, CA, in 1986, 1989 and 1989 respectively.
¿From 1982 to 1984, he was a Research and Teaching Assistant at the State University of Gent on a fellowship from the National Science Foundation, Belgium. In 1984, he received a Fulbright grant and went to Stanford University, where he held appointments as a Research and Teaching Assistant. From 1989 until 1991 he was a Member of the Scientific Staff of the Philips Research Laboratory Belgium. He currently is an assistant professor in the mobile communications department of the Eurecom Institute, Sophia Antipolis, France. His present areas of interest include the design and implementation of efficient adaptive filtering techniques and their application to electrical and acoustic echo cancellation, blind equalization, spatial diversity techniques and antannea arrays, multiuser interference problems and mobile radio receiver design.

**Karim Maouche** was born in Algiers, Algeria, on May 1, 1967. He received the degree of Engineer in Control from "Ecole Nationale des Ingénieurs et Techniciens d'Algérie" in 1989 and the "Diplome d'études approfondies" in Control and Signal Processing in 1992 from the "Laboratoire des Signaux et Systèmes, Univesité de Paris-Sud" . He is currently a teaching and research assistant in the mobile communications department of the Eurecom Institute, Sophia Antipolis, France, where he is working towards the Ph.D. degree on the subject of acoustic echo cancellation. His current research interests are efficient adaptive filtering techniques applied to acoustic echo cancellation.