A Modular Prewindowing Framework for Covariance FTF RLS Algorithms[†]

Dirk T.M. Slock[‡]

Institut EURECOM, 2229 route des Crêtes, Sophia Antipolis,
06560 Valbonne, France.

Thomas Kailath

Information Systems Laboratory, Department of Electrical Engineering,
Stanford University, Stanford CA 94305, USA.

Please send all correspondence to:

Dirk Slock

Leemstraat 14
B-9080 Zaffelare
Belgium

[†]This work was supported in part by the Joint Services Program at Stanford University (US Army, US Navy, US Air Force) under Contract DAAL03-88-C-0011 and the SDI/IST Program managed by the Office of Naval Research, under Contract N00014-85-K-0550.

[‡]Dirk Slock was with the Information Systems Laboratory at Stanford University until September 1989.

Number of text pages: 18

Number of tables: 3

Number of figures: 1

Proposed running headline: Modular Prewindowing FTF Framework

Keywords: Recursive Least-Squares, Fast Transversal Filter algorithm, prewindowing, framework, covariance window, sequential processing, modularity, numerical stability, square-root algorithm.

A Modular Prewindowing Framework for Covariance FTF RLS Algorithms

Dirk T.M. Slock and Thomas Kailath

Abstract

In a companion paper [16], a Fast Transversal Filter (FTF) algorithm was derived for solving *multichannel multiexperiment* Recursive Least-Squares (RLS) problems arising in adaptive FIR filtering. By introducing *sequential processing* of the different channels and experiments, the multichannel multiexperiment algorithm was decomposed into a set of intertwined single-channel single-experiment algorithms, resulting in a *modular* algorithm structure. The algorithm was derived under the *prewindowing* assumption. However, using an *embedding* into multichannel and multiexperiment problems, we show how the conventional FTF algorithms for the Growing-Window and Sliding-Window *Covariance* cases follow naturally from the modular prewindowed algorithm. Furthermore, taking the sequential processing one step of granularity further, we derive modular multichannel FTF algorithms for these covariance cases also.

1 Introduction

Two popular classes of algorithms for adaptive FIR filtering [5] are the Least-Mean-Square (LMS) algorithm, which is based on a stochastic-gradient approach, and the Recursive Least-Squares (RLS) algorithm, which minimizes a deterministic sum of squared errors. RLS algorithms are usually made adaptive by introducing exponential weighting in the sum of squared errors, to dampen the effect of data in the remote past (making the effective window length finite). Exploiting the shift relation between two consecutive regression vectors, which is typical of the adaptive FIR filtering problem, so-called *fast RLS* algorithms can be derived with a computational complexity of $O(N)$ (N denotes the filter order), the same order of complexity as for the LMS algorithm.

The simplest fast RLS algorithms are obtained by introducing the *prewindowing* assumption [5] (in which all data before time zero are assumed to be zero). The *covariance* (or unwindowed) method [12],[4] is an alternative approach in which no assumptions are made on unavailable data. So the covariance method only starts considering regression vectors from the moment they can be completely filled with available data. Covariance methods can perform noticeably better than prewindowed algorithms for short window lengths. They come in two varieties, the *growing-window* and the *sliding-window* covariance algorithms, (often considered) as alternatives for prewindowed algorithms without or with exponential weighting respectively. The derivation of fast covariance algorithms has always been somewhat of an art, in that their derivation requires additional insight which goes still a step beyond that required for prewindowed algorithms. In [8], a unified geometric theory was presented which tries to cover the derivation of all fast RLS algorithms. However, only the prewindowed case falls out nicely from this theory. In another attempt at unification, the two covariance cases were imbedded into prewindowed problems in [13],[10]. However, the algorithms resulting from the prewindowed embedding did not coincide with the existing covariance algorithms, and were in fact computationally more complex.

We shall also consider the prewindowing embedding framework of [13],[10] and we shall concentrate on the Fast Transversal Filter (FTF) RLS algorithm (in [13],[10], Fast Lattice RLS algorithms were considered). In contrast to the results obtained in [13],[10], applying the mod-

ular prewindowed FTF algorithm of [16] to the embedded problems will enable us to recover the existing covariance algorithms, modulo the numerical stabilization part which has only recently been introduced in FTF algorithms [18]. Apart from providing a unified prewindowing framework for the derivation of covariance algorithms, this approach will also enable us to straightforwardly extend the numerical stabilization from prewindowed to covariance algorithms. Then taking the modularity one level of granularity further, we shall straightforwardly obtain *modular multichannel* covariance algorithms (see [5] or [16] for an introduction to multichannel adaptive FIR filtering problems). Exponential weighting will be superimposed on all windowing schemes considered, and some numerical consequences of this will be discussed.

This paper is organized as follows. For the notation used in the prewindowed multichannel multiexperiment framework, we refer to [16]. In section 2, we introduce some extra notation that is pertinent to the covariance method. In section 3, we describe the embedding of the growing-window covariance problem into a prewindowed problem with one extra artificial channel, followed by the algorithm derivation. The two-experiment prewindowed embedding and the algorithm derivation for the sliding-window covariance method are described in section 4. In section 5, we describe the customization to the covariance algorithms of certain details in the computational redundancies and error feedback that are used for the stabilization of the propagation of roundoff errors in FTF algorithms. Finally in section 6, some concluding remarks are given.

2 Problem Formulation and Notation

Again, for the formulation of and the notation used in the prewindowed multichannel multiexperiment framework, we refer to [16]. We shall derive modular covariance FTF algorithms for the case of a single experiment, but multiple (p) channels with possibly different filter lengths in each channel (and we shall drop the experiment subscript 1 from the notation for covariance quantities). This case is perhaps of most interest in practical applications. We shall embed the growing- and sliding-window methods of the covariance formulation into various prewindowed problems, and will thus be able to apply the general modular algorithm of [16] to derive modular covariance FTF algorithms in a systematic way and with little extra effort. Given the frame-

work presented here, it will be possible to straightforwardly derive other modular covariance algorithms than the ones presented here (e.g. involving multiple experiments), should the need for such algorithms arise.

In the prewindowed algorithm of [16], we considered vectors of length qM and strictly speaking, one should take $M = \infty$ (as time proceeds, the number of samples can exceed any finite number). So we are working with semi-infinite strings of data which have only one endpoint at which something happens as time proceeds. Now at time T, we do not have the infinite past available in general, and so the way the prewindowing method gets around the infinity issue is by assuming that all data before time $t = 0$ are zero (only in the case this assumption coincides with reality, we do have the infinite past available). Using the prewindowing assumption, the effective number of terms in the LS cost function reduces from infinity to T+1. In the covariance method on the other hand, no extra assumptions on the data are made and the entries in all regression vectors $X_N(T)$ used in the cost function are actual available data. Since in practice, we can only have a finite window on the data at our disposal, strings of used data are now marked by two endpoints.

Strictly speaking, it would be possible to describe the covariance algorithms with the notation introduced in [16], as will soon become clear. However, one will get more insight from considering this notation next to the conventional covariance notation (see [2],[4]) with its specific meaning. We shall introduce this conventional notation here also and find interesting interpretations by comparing both points of view. For a windowlength equal to L, the LS cost function for the covariance method is

$$\xi_{N,L}(T) = \min_{W_{N,L,T}} \sum_{t=T-L+1}^{T} \lambda^{T-t} \left\| d(t) + W_{N,L,T} X_N(t) \right\|^2 \quad . \tag{1}$$

We can reformulate the LS problem in a L-dimensional vector space in which we shall consider vectors of the form[1]

$$x_{j,L,T} \stackrel{\triangle}{=} [x_j(T) \;\; x_j(T-1) \cdots x_j(T-L+1)]^H \tag{2}$$

with similar definitions for other vectors and data matrices. The weighting matrix appearing in the inner product $< x_L, y_L >_\Omega \stackrel{\triangle}{=} x_L^H \Omega y_L$ will be[2] $\Omega = \Lambda_L \stackrel{\triangle}{=} \bigoplus_{k=0}^{L-1} \lambda^k$. In this notation, we can

[1] In this paper, superscript H denotes Hermitian (complex conjugate) transpose.

[2] The Kronecker product $\otimes$ of two matrices is defined as $A \otimes B \stackrel{\triangle}{=} [a_{ij}B]$, a block matrix in which block (i, j)

write

$$W_{N,L,T} = -d^H_{L,T}\Lambda_L K_{N,L,T} \quad , \quad \epsilon_{N,L,T} = P^{\perp}_{N,L,T} d_{L,T} \quad , \quad \xi_{N,L}(T) = \epsilon^H_{N,L,T}\Lambda_L \epsilon_{N,L,T} \quad . \qquad (3)$$

We will invariably denote the pinning vector by σ, the shift matrix by S, and the weighting matrix by Ω, regardless of whether we are considering the Mq-dimensional vector space of the prewindowed formulation of [16], or the L-dimensional vector space of the covariance formulation considered here.

In the covariance algorithms, we shall need to pin down data at the other endpoint of the window as well, and hence we introduce a second $L \times 1$ pinning vector $\pi \stackrel{\triangle}{=} [0 \cdots 0\ 1]^H$ with the property $x^H_{j,L,T}\pi = x_j(T-L+1)$. Note that if we introduce the exchange matrix J with ones on the antidiagonal and zeros elsewhere, then we have $\pi = J\sigma$ and the shift matrix corresponding to π is $S^H = JSJ$. So the duality between both pinning mechanisms is modulo J. Note that $R_{S^H X_{N,L,T};S^H\Lambda_L S} = \lambda R_{N,L-1,T-1}$ and $R_{SX_{N,L,T};S\Lambda_L S^H} = R_{N,L-1,T}$. We can introduce a dual Kalman gain D_X and associated likelihood variable δ_X, which can be defined (just as γ_X) as both a prediction error and an error covariance. The dual version of equations [16]-I-(1-10) (equations (1) through (10) in Table I in [16]) and [16]-(23) holds and one can find it by making the substitutions $(\sigma, S, C, \gamma) \leftrightarrow (\pi, S^H, D, \delta)$. The predicted residuals $(.)^p$ in [16]-I-(5) are now backwards predicted, instead of forwards, and we shall therefore call them smoothed residuals and denote them by $(.)^s$ (as in [4]).

The single-experiment, modular multichannel covariance quantities are described in Table I. Note that because of our compact algorithm description via f_U, f_D and f_J (see [16]), it is strictly speaking not necessary to name all forward and backward prediction errors arising in the algorithm. In Table I, we only give a detailed account of the error signals arising in the joint-process filtering. The description and interpretation of the prediction errors arising in the forward and backward prediction problems runs totally parallel.

Suitable position for Table I

---

is $a_{ij}B$. The direct sum $\oplus$ of matrices is defined as $\bigoplus_{k=1}^{m} A_k = A_1 \oplus A_2 \oplus \cdots \oplus A_m \stackrel{\triangle}{=}$ block-diag $\{A_1, A_2, \ldots, A_m\}$.

3 The Growing-Window Covariance (GWC) FTF algorithm

The GWC method considers a window length that is growing with time, as in the prewindowing method. However, the window only starts at the first point in time when the regression vector $X_N(T)$ can be completely filled up with available data. Let $N_m \stackrel{\triangle}{=} \max_{1 \le j \le p} N_j$ be the largest filter order in the different channels. Assuming that at time T we can dispose of the data in $[0, T]$, the GWC method takes $L = T - N_m + 2$.

3.1 Embedding into a prewindowed problem with one extra channel

Consider the following artificial $(p+1)^{\text{st}}$ channel with input signal $y(t) = \delta(t+1)$, a unit pulse at time $t = -1$. Let $\overline{N} = N + N_m$. Then considering the augmented prewindowed data matrix, we have

$$X_{\overline{N},T} \stackrel{\triangle}{=} [X_{N,T}\ Y_{N_m,T}] \sim \left[P^{\perp}_{Y_{N_m,T}} X_{N,T} \quad Y_{N_m,T}\right] = \left[\begin{bmatrix} X_{N,L,T} \\ 0 \end{bmatrix} \quad Y_{N_m,T}\right] \tag{4}$$

where the equivalence relation "$\sim$" means "has the same column space as". Or more explicitly (for $p = 1$ and omitting the channel subscript),

$$\left[\begin{array}{cccc|cccc} x(T) & x(T-1) & \cdots & x(T-N+1) & & & & \\ x(T-1) & x(T-2) & \cdots & x(T-N) & & & & \\ \vdots & & & \vdots & & \mathbf{0} & & \\ x(N-1) & \cdots & x(1) & x(0) & & & & \\ \hline x(N-2) & \cdots & x(0) & 0 & 0 & \cdots & 0 & 1 \\ \vdots & \iddots & 0 & 0 & \vdots & \iddots & \iddots & 0 \\ x(0) & \iddots & & \vdots & 0 & 1 & \iddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{array}\right] \tag{5}$$

$$\sim \left[\begin{array}{cccc|cccc} x(T) & x(T-1) & \cdots & x(T-N+1) & & & & \\ x(T-1) & x(T-2) & \cdots & x(T-N) & & & & \\ \vdots & & & \vdots & & \mathbf{0} & & \\ x(N-1) & \cdots & x(1) & x(0) & & & & \\ \hline 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 1 \\ \vdots & \iddots & 0 & 0 & \vdots & \iddots & \iddots & 0 \\ 0 & \iddots & & \vdots & 0 & 1 & \iddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{array}\right]$$

where we have omitted the extra rows of zeros at the bottom. Since we now have an orthogonal decomposition of the columnspace of $X_{\overline{N},T}$, we can additively decompose the projection operator

$$P_{\overline{N},T} = \begin{bmatrix} P_{N,L,T} & 0 \\ 0 & 0 \end{bmatrix} + P_{Y_{N_m,T}} \tag{6}$$

or hence

$$P^{\perp}_{\overline{N},T} = \begin{bmatrix} P^{\perp}_{N,L,T} & & 0 \\ & 0_{N_m} & \\ 0 & & I \end{bmatrix} \quad . \tag{7}$$

This means that working with the columnspace of $X_{\overline{N},T}$ is equivalent to working with the columnspace of $X_{N,L,T}$. In terms of the filter operator K, we have

$$K_{\overline{N},T} = [K_{N,L,T} \mid * \,] \ , \qquad T \geq N_m \tag{8}$$

where "$*$" denotes "don't care". Indeed, for $T \geq N_m$, the part $Y_{N_m}(T)$ of the regression vector $X_{\overline{N}}(T) = \left[X^H_N(T)\ Y^H_{N_m}(T)\right]^H$ will contain only zeros. Hence, $K_{\overline{N},T}$ applied to all of $X_{\overline{N}}(T)$ will produce the same output as $K_{N,L,T}$ applied to only $X_N(T)$. Actually, the "$*$" entries in (8) can easily be determined. They serve to produce zeros in the entries of an error vector $P^{\perp}_{\overline{N},T}Y$ that correspond to the nonzero rows in $Y_{N_m,T}$ (consider the zero diagonal block 0_{N_m} in the expression for $P^{\perp}_{\overline{N},T}$ in (7)). So $K_{\overline{N},T}X_{\overline{N}}(k-2) = 0, \ \ k = 1, \ldots, N_m$, and hence

$$*^k = K^{N+k}_{\overline{N},T} = -K_{N,L,T}X_N(k-1) \ \ , \qquad k = 0, \ldots, N_m - 1 \tag{9}$$

where the "$*$" in (8) has been decomposed as $* = \left[*^0 \ *^1 \cdots *^{N_m-1}\right]$.

3.2 Algorithm Derivation

With the above embedding, we can apply the modular multichannel prewindowed FTF algorithm with $p+1$ channels to arrive at a modular multichannel GWC FTF algorithm for p channels, which is described in Table II. Comparing the notation for the modular p-channel GWC problem with that of the equivalent modular $(p+1)$-channel prewindowed problem, we

find

$$
\begin{aligned}
1 \le j \le p: \begin{cases} A_{j,\overline{N},T}=[A_{j,N,L-1,T} \mid *] & \alpha_{j,\overline{N}}(T)=\alpha_{j,N,L-1}(T) \\ B_{j,\overline{N},T}=[B_{j,N,L-1,T} \mid *] & \beta_{j,\overline{N}}(T)=\beta_{j,N,L-1}(T) \\ \tilde{C}_{j,\overline{N},T}=\left[\tilde{C}_{j,N,L-1,T} \mid * \right] & \gamma_{j,\overline{N}}(T)=\gamma_{j,N,L-1}(T) \end{cases} \\
j=p+1: \begin{cases} A_{p+1,\overline{N},T}=[0_{1\times N} \mid 1\ 0_{1\times N_m}] & \alpha_{p+1,\overline{N}}(T)=\lambda^{T+1} \\ B_{p+1,\overline{N},T}=[D_{N,L,T} \mid *] & \beta_{p+1,\overline{N}}(T)=\delta_{N,L}(T) \\ \tilde{C}_{p+1,\overline{N},T}=\left[\tilde{C}_{N,L,T} \mid * \right] & \gamma_{p+1,\overline{N}}(T)=\gamma_{N,L}(T) \end{cases} \\
W_{\overline{N},T}=[W_{N,L,T} \mid *] \qquad \xi_{\overline{N}}(T)=\xi_{N,L}(T)
\end{aligned} \tag{10}
$$

We see that the prediction problem for channel $p+1$ degenerates since the forward prediction part becomes trivial. The backward predictor corresponds to the backward Kalman gain of the covariance algorithm. Note that the 1 entry of the backward predictor $B_{p+1,\overline{N},T}$ appears in the "$*$" portion (namely at position $\overline{N}$). However, for the stable operation of the FTF algorithm and particularly the operator f_D (see section 5), it is important that this entry is included in the filter. This is especially important for the corresponding entry $\tilde{C}^{\overline{N}}_{p+1,\overline{N}+1,T}$ of the order-updated Kalman gain. Therefore, we have embedded the filters for channel $p+1$ into vectors with one extra position, as indicated in II-(4), without modifying the basic operation of the order downdate. The only nontrivial output from the order update part in channel $p+1$ is $\tilde{C}^{\overline{N}}_{p+1,\overline{N}+1,T}$. We can calculate this entry separately as follows. We have for the order-updated Kalman gain

$$
C_{p+1,\overline{N}+1,T} = [C_{p,N,L-1,T} \mid *] \ , \quad \gamma_{p+1,\overline{N}+1}(T) = \gamma_{p,N,L-1}(T) \ . \tag{11}
$$

Combining this with (9) gives II-(3). So the order-update part of channel $p+1$ is reduced to just II-(3).

Suitable position for Table II

Note that upon putting the feedback coefficients $K_k = 0$ in the definition of the operator f_D (see [16]) and omitting II-(5), the algorithm of Table II for $p = 1$ corresponds exactly to the GWC FTF algorithm that can be found in [2] (the algorithm in [4] is similar, but uses so-called overnormalized filters). There are some differences in the computation of error covariances, but those are non-essential. There is also a minor difference in notation in that the covariance algorithms of [2],[4] assume $x(1-N_m)$ to be the first available sample and start at time 0, whereas we assume $x(0)$ to be the first sample and hence start at time $N_m - 1$. With

$p = 1$, the algorithm of Table II even gives the conventional multichannel GWC FTF algorithm if the "scalar" signal $x_1(.)$ is appropriately defined as a vector of signals (as in [3],[4]). This "vectorization" approach is straightforward for multichannel problems with the same filter order in each channel. To obtain a multichannel covariance FTF algorithm in the conventional style for the case of unequal filter orders in the different channels, one can again use $p = 1$ in Table II, and replace the simple permutation matrices $\mathcal{P}_0$ and $\mathcal{P}_1$ by the composite permutation matrices appearing in [16]-(27),[16]-(28) (see [15] also).

3.3 Initialization

Using the prewindowing embedding, we can employ the initialization of the modular prewindowed FTF algorithm. So consider the following noncausal part of the signals:
$x_j(t) = \lambda^{-(N_m+1)/2}\mu_j\delta\left(t+1+j+\Sigma_j\right), \quad j = 1,\ldots,p \ , \ t < 0$ (we have advanced the noncausal parts one time unit compared to the usual prewindowed algorithm, since for the GWC embedding, channel $p+1$ starts at time $t=-1$ instead of time zero, and we have scaled μ_j by a power of λ to simplify the expressions for the inital values in (12) below). Note that to consider some noncausal part in the signals $x_j(.)$, we have to change the definition of vectors like $x_{j,L,T}$ (see (2)). However, such changes are obvious and we shall not consider them in detail here (see [2],[4] for details). Now, since channel $p+1$ is just artificial, we have put $\mu_{p+1} = 0$. But this implies that $R_{\overline{N},T}$ does not become full rank until $T = N_m - 2$. Since $\tilde{C}_{\overline{N},T}$ uses $R_{\overline{N},T-1}$ in its definition, we shall consider initializing all quantities at time $T = N_m - 1$, so that we can start the algorithm of Table II at time $T = N_m$. Using the definitions of Table I, one can straightforwardly obtain the following initial values

$$
\begin{aligned}
&A_{j,N,0,N_m-1} = [0\cdots 0 \ \ 1]\,\mathcal{P}_{j-1} \ , && \alpha_{j,N,0}\,(N_m-1) = \lambda^{j-1+\Sigma_j}\mu_j\mu_j^H \\
&B_{j,N,0,N_m-1} = [0\cdots 0 \ \ 1]\,\mathcal{P}_{j} \ , && \beta_{j,N,0}\,(N_m-1) = \lambda^{j-1+\Sigma_{j-1}}\mu_j\mu_j^H \qquad j = 1,\ldots,p \\
&\tilde{C}_{0,N,1,N_m} \ = \ -X_N^H\,(N_m-1)\,\lambda^{-1}R^{-1}_{N,N_m-2} \ , && \gamma^{-1}_{0,N,1}\,(N_m) \ = \ 1 - \tilde{C}_{0,N,1,N_m}X_N\,(N_m-1) \\
&D_{N,1,N_m-1} = \gamma_{0,N,1}(N_m)\tilde{C}_{0,N,1,N_m} \ , && \delta_{N,1}\,(N_m-1) = \gamma_{0,N,1}\,(N_m) \\
&(W_{N,1,N_m-1}, \epsilon_{N,1}(N_m-1)) \ = \ f_J\left(W_0, \tilde{C}_{0,N,1,N_m}, \gamma_{0,N,1}(N_m), d(N_m-1), X_N(N_m-1)\right)
\end{aligned}
\tag{12}
$$

where

$$
R_{N,N_m-2} \ = \ \bigoplus_{j=1}^{p}\left\{\lambda^{j-1+\Sigma_{j-1}}\left[\bigoplus_{k=1}^{N_j}\lambda^{N_j-k}\right]\otimes\mu_j\mu_j^H\right\} \tag{13}
$$

is an easily invertible block-diagonal matrix.

4 The Sliding-Window Covariance (SWC) FTF algorithm

The SWC method considers a fixed window length L. It offers an alternative to the prewindowed method with exponential weighting for achieving a finite effective window-length. In [11], it was shown that under certain conditions both windowing methods have an identical performance for corresponding effective window-length. However, this equivalence only holds asymptotically for large window-lengths. For short windows, the SWC method might still be preferable for certain types of nonstationarities since its window really cuts off the past beyond time $T - L$.

4.1 Embedding into a two-experiment prewindowed problem

It is easy to see that the cost function for the SWC problem is related to the cost functions of two related prewindowed problems with a similar relationship for the sample covariance matrices, viz.

$$\begin{aligned} \|\epsilon_{N,L,T}\|^2 &= \|\epsilon_{N,T}\|^2 - \lambda^L \|\epsilon_{N,T-L}\|^2 \\ R_{N,L,T} &= R_{N,T} - \lambda^L R_{N,T-L} \end{aligned} \tag{14}$$

When time progresses from $T-1$ to T, the sample covariance matrix $R_{N,L,T-1}$ now undergoes a rank two modification :

$$R_{N,L,T} = \lambda R_{N,L,T-1} + X_N(T)X_N^H(T) - \lambda^L X_N(T-L)X_N^H(T-L) \ . \tag{15}$$

Hence, an embedding of the SWC problem into a two-experiment prewindowed problem comes naturally. Specifically, consider

$$\begin{cases} x_{1,j}(T) = x_j(T) \\ x_{2,j}(T) = x_j(T-L+1) \end{cases} \qquad \begin{cases} \omega_1 = 1 \\ \omega_2 = -\lambda^{L-1} \end{cases} \tag{16}$$

and similarly for the desired-response signal. Note that we use a window of length $L-1$ for reasons that will soon become clear. Note also that here we have an example of an indefinite weighting ω, but the cost function $\xi_{N,L-1}(T)$ is nevertheless positive definite since $R_{N,L-1,T}$ is (assumed to be (for $L > N$)) a positive definite matrix.

4.2 Algorithm Derivation

With the above embedding, we can apply the modular multiexperiment multichannel prewindowed FTF algorithm to arrive at a modular multichannel SWC FTF algorithm, which is described in Table III. Comparing the notation for the modular p-channel SWC problem with that of the equivalent modular prewindowed problem, we find for (A, α)

$$1 \leq j \leq p : \begin{cases} A_{1,j,N,T}{=}A_{j,N,L,T} & \alpha_{1,j,N}(T){=}\alpha_{j,N,L}(T) \\ A_{2,j,N,T}{=}A_{j,N,L-1,T} & \alpha_{2,j,N}(T){=}\alpha_{j,N,L-1}(T) \end{cases} \tag{17}$$

and similarly for (B, β) and (W, ξ). For the Kalman gains and likelihood variables, we find

$$\begin{aligned} & \begin{cases} C_{1,j,N,T} = & -X^H_{j,N}(T) R^{-1}_{j,N,L,T} & = C_{j,N,L,T} \\ \tilde{C}_{1,j,N,T} = & -X^H_{j,N}(T) \lambda^{-1} R^{-1}_{j,N,L-1,T-1} & = \tilde{C}_{j,N,L,T} \end{cases} \\ \Rightarrow \quad & \gamma_{1,j,N}(T) \;=\; \gamma_{j,N,L}(T) \\ & \begin{cases} C_{2,j,N,T} = \lambda^{L-1} X^H_{j,N}(T{-}L{+}1) R^{-1}_{j,N,L-1,T} & = -\lambda^{L-1} \tilde{D}_{j,N,L,T} \\ \tilde{C}_{2,j,N,T} = \quad -X^H_{j,N}(T{-}L{+}1) R^{-1}_{j,N,L,T} & = \lambda^{1-L} D_{j,N,L,T} \end{cases} \\ \Rightarrow \quad & \gamma^{-1}_{2,j,N}(T) \;=\; -\lambda^{-2(L-1)} \delta_{j,N,L}(T) \end{aligned} \tag{18}$$

for $j = 1, \ldots, p$. In the second experiment, it is interesting to note the correspondence between γ^{-1} and δ and the ensuing reversed role of the normalization as displayed in the correspondence between $(C, \tilde{C})$ and $(\tilde{D}, D)$. It clearly is convenient to introduce the following scaled quantities

$$\widehat{D}_{j,N,L,T} \stackrel{\triangle}{=} \lambda^{1-L} D_{j,N,L,T}, \quad \widehat{\tilde{D}}_{j,N,L,T} \stackrel{\triangle}{=} \lambda^{L-1} \tilde{D}_{j,N,L,T}, \quad \widehat{\delta}_{j,N,L}(T) \stackrel{\triangle}{=} \lambda^{-2(L-1)} \delta_{j,N,L}(T) \tag{19}$$

which can also be found from [16]-I-(1,2,3) by replacing π by $\hat{\pi} \stackrel{\triangle}{=} \lambda^{1-L} \pi$ in the definition of the unscaled quantities.

Suitable position for Table III

It can be seen from III-(7,8) that the algorithm produces both $W_{N,L-1,T}$ and $W_{N,L,T}$. Hence, an SWC algorithm with a window of length either $L-1$ or L will produce $W_{N,L,T}$. However, only the choice $L-1$ will produce $\epsilon_{N,L}(T)$, since this error signal only comes about in the update of the window-length (processing experiment 1). When we omit III-(5,6), the algorithm of Table III for $p = 1$ and $\lambda = 1$ corresponds again exactly to the SWC FTF algorithm that can be found in [2]. Also the other comments that we made on the comparison of GWC FTF algorithms apply here to the SWC case.

4.3 Initialization

Two strategies are possible: a prewindowing initialization, which will have a prewindowing effect for a duration of $L + N_m - 1$ samples (in the SWC method, the emphasis usually is not on initialization effects, but on the rectangular window shape in steady-state operation), or a GWC initialization, enforcing the covariance character from the very start. In the first strategy, we can readily apply the general initialization methodology of [16] to arrive at

$$\begin{aligned}
A_{j,N,L-1,-1} &= [0 \cdots 0 \;\; 1]\, \mathcal{P}_{j-1} \;, \quad \alpha_{j,N,L-1}(-1) = \lambda^{j-1+\Sigma_j}\zeta_j \\
B_{j,N,L-1,-1} &= [0 \cdots 0 \;\; 1]\, \mathcal{P}_{j} \;, \qquad \beta_{j,N,L-1}(-1) = \lambda^{j-1+\Sigma_{j-1}}\zeta_j \qquad j = 1, \ldots, p \\
\tilde{C}_{0,N,L,0} &= [0 \cdots 0] \;, \qquad\qquad \gamma_{0,N,L}(0) = 1 \\
\widehat{D}_{0,N,L,0} &= [0 \cdots 0] \;, \qquad\qquad \widehat{\delta}_{0,N,L}(0) = \lambda^{1-L} \\
W_{N,L-1,-1} &= W_0
\end{aligned} \tag{20}$$

where $\zeta_j \stackrel{\triangle}{=} \mu_j \omega \mu_j^H$ are some positive constants. This initialization allows one to start the SWC algorithm of Table III at time $T = 0$. In the second strategy, one starts to run the GWC algorithm of Table II and lets the window-length grow until time $T = L + N_m - 2$ at which point the desired length L is reached, and then one switches to the SWC algorithm of Table III, starting at time $T = L + N_m - 1$. At time $T = L + N_m - 2$, one will have to scale the quantities D, δ to $\widehat{D}$, $\widehat{\delta}$, but there is time for that since the GWC algorithm takes less computation per time step than the SWC algorithm.

5 Numerical Considerations

We refer to [16] for a description of the introduction of redundancies and error feedback for the stabilization of the propagation of round-off errors in the general modular prewindowed FTF algorithm. Here we describe some specific details that are particular for the covariance algorithms.

5.1 The GWC Covariance Algorithm

In the GWC algorithm, the two ways of computing the backward prediction error in channel p+1 correspond to the two ways of computing the "cross-likelihood" quantity $\eta_{N,L}(T)$ as indicated in Table I. The numerical stabilization mechanism introduced in [18] and extended in [16] for

the general modular FTF algorithm works best in a stationary environment. However, the impulse in channel $p+1$ of the multichannel prewindowed embedding of the GWC algorithm represents anything but a stationary signal. Indeed, the whole purpose of this channel is to take care of some initial conditions, whose influence is very much of a transient nature. Especially in the presence of exponential weighting, the quantities $\delta_{N,L}(T)$ and $D_{N,L,T}$ decay exponentially as λ^T. Therefore the influence of these quantities on the rest of the algorithm state becomes negligible after a few time constants $\frac{1}{1-\lambda}$, at which point one could put these quantities equal to zero, which is equivalent to switching to the (p-channel) prewindowed algorithm.

More precisely, the quantity $\delta_{N,L}(T)$ behaves as $\delta_{N,L}(T) = \lambda^{L-1} + O\left(\lambda^{2L}\right)$, where $L = T - N_m + 2$ in the context of the GWC algorithm. Now, the open-loop ($K_1 = 0$) eigenvalue associated with the error propagation for $D_{N,L,T}$, the backward prediction filter for channel $p+1$, is given by $\frac{\delta_{N,L}(T)}{\lambda\delta_{N,L-1}(T-1)}$. In a stationary channel, this value would average out to $\frac{1}{\lambda}$, the familiar unstable mode. However, because of the exponential decay of $\delta_{N,L}(T)$, the open-loop eigenvalue averages out to 1! Hence it is very easy for the feedback loop (involving K_1) to stabilize this marginal instability and we can conclude that the nonstationarity in channel $p+1$ is actually beneficial for its numerical behavior.

However, II-(5) reveals a problem that renders the algorithm as presented in Table II not quite amenable to a practical implementation. Indeed, the quantity λ^{-T}, which diverges to infinity, has to be multiplied with $\delta_{N,L}(T)$, which converges to zero. This becomes a numerically ill-conditioned operation as time grows (not to mention representation problems). The GWC FTF algorithm presented in [4] suffers from the same problem. It is clearly desirable to work instead with the scaled quantity

$$\check{\delta}_{N,L}(T) \stackrel{\triangle}{=} \lambda^{-(L-1)}\delta_{N,L}(T) \;=\; 1 - O\left(\lambda^{L}\right) \tag{21}$$

which is initialized as $\check{\delta}_{N,1}(N_m - 1) = \delta_{N,1}(N_m - 1) < 1$ and converges to one. Introducing $\check{\delta}_{N,L}(T)$ into the algorithm requires some changes in the handling of channel $p+1$. The following

rearrangement leads to a mere change of the feedback coefficients in f_D. Replace II-(4,5) by

$$\begin{aligned}
&\text{II}-(4'): \quad \left([D_{N,L,T} \quad 1], \check{\delta}_{N,L}(T), \left[\tilde{C}_{N,L,T} \quad 0\right], \gamma^{-s}_{N,L}(T)\right) \\
&\quad = f_{D'}\left([D_{N,L-1,T-1} \quad 1], \check{\delta}_{N,L-1}(T-1), \left[\tilde{C}_{p,N,L-1,T} \quad \tilde{C}^{\overline{N}}_{p+1,\overline{N}+1,T}\right], \gamma^{-s}_{p,N,L-1}(T), \begin{bmatrix} X_N(T) \\ 0 \end{bmatrix}\right) \\
&\text{II}-(5'): \quad \gamma_{N,L}(T) \;=\; \gamma^{a}_{N,L}(T) \;=\; \lambda^N \check{\delta}_{N,L}(T) \prod_{j=1}^{p} \left(\beta_{j,N,L-1}(T)\alpha^{-1}_{j,N,L-1}(T)\right)
\end{aligned} \tag{22}$$

The downdate operator $f_{D'}$ is defined as f_D in Table [16]-IV except for the following changes related to K_1, K_2

$$\begin{aligned}
& r^{(1)}_N(T) \;=\; r^{f}_N(T) \;=\; r^{pf}_N(T)\,\gamma_N(T) \\
& \beta_N(T) \;=\; \beta_N(T-1) + r^{f}_N(T) r^{ps\,H}_N(T) \\
& \text{if } \; 1-\check{\delta}_{N,L}(T) < \epsilon^{mp} \;\; \text{ then} \\
& \qquad D_{N,L,T} = 0 \\
& \qquad \check{\delta}_{N,L}(T) = 1 \\
& \text{end if}
\end{aligned} \tag{23}$$

where ϵ^{mp} is the machine precision. The above changes correspond to taking $K_1 = 1$, which is clearly sufficient here for stabilizing $\Delta D_{N,L,T}$. The error propagation associated with $\check{\delta}_{N,L}(T)$ on the other hand is exponentially unstable. From (22),(23), we get

$$\frac{\Delta\check{\delta}_{N,L}(T)}{\Delta\check{\delta}_{N,L-1}(T-1)} \;=\; \frac{\check{\delta}_{N,L}(T)}{\check{\delta}_{N,L-1}(T-1)} \;=\; 1-D_{N,L-1,T-1}X_N(T)\tilde{C}^{\overline{N}H}_{p+1,\overline{N}+1,T} \;=\; 1+O\left(\lambda^T\right) \;>\; 1\;. \tag{24}$$

However, since $\check{\delta}_{N,L}(T)$ does not have a constant average value during the transient period considered here, it is more relevant to consider the relative errors. And (24) leads immediately to $\Delta \log \check{\delta}_{N,L}(T) = \Delta \log \check{\delta}_{N,L-1}(T-1)$. So we have a random walk for the relative errors, a mild instability. The quantity $\check{\delta}_{N,L}(T)$ increases to one during the transient period. Due to round-off errors, the built-in switch to the prewindowed algorithm (see (23)) may occur a bit prematurely, but the effect of this will be negligible for reasonable values of ϵ^{mp}. Simulation experience with the algorithm indicates a stable behavior.

In the case of $\lambda = 1$, the quantities $\delta_N(T)$ and $D_{N,T}$ decay as $\frac{1}{T}$. In this case however, the numerical error propagation shows a random walk behavior (digital integrator, see [18]), irrespective of the feedback coefficient values. So $\lambda < 1$ is desired for numerical stability.

5.2 The SWC Covariance Algorithm

With the feedback coefficients proposed in [16]-(44), the range of stable operation for λ decreases as the disparity of the weights ω_i in the different experiments increases. This remark is especially relevant for the SWC algorithm with its indefinite weighting coefficients $\omega_1 = 1$ and $\omega_2 = -\lambda^{L-1}$. With an exponential weighting factor λ present, the following approximation will often hold: $\sum_{i=1}^{2} \omega_i = 1-\lambda^{L-1} \approx 1$. This leads us to consider the following modification to [16]-(44)

$$K_{1,i,j}(T) \;=\; K_{2,i,j}(T) - 1 \;=\; \frac{0.75}{\omega_i} \frac{\gamma_{i,j,N}(T)}{\gamma_{i,j,N+1}(T)} \tag{25}$$

where the γ-ratio could be omitted. Note especially the alternating sign of K_1, K_2-1 as the experiments get processed. However, the closeness of the choices in [16]-(44) and (25) will get lost as λ approaches unity. In particular, we get for $\lambda = 1$ that $\sum_{i=1}^{2} \omega_i = 0$! This reflects the fact that the average value of the sample covariance matrix becomes zero for such a choice of weighting coefficients in a general two-experiment case (with identical signal statistics in the two experiments). However, in the SWC case, there is a strong correlation between the signals in the two experiments (one being a delayed version of the other). So the average sample covariance matrix will not be zero but proportional to $L-1$. Because of this, it is desirable to let the feedback coefficients K_1, K_2-1 not go to zero as λ approaches one, but to some finite value, for instance the value provided in (25). With the choice of feedback coefficients given in (25), preliminary simulation experience indicates stable operation for $\lambda \in (1-\frac{1}{2N}, 1]$ when L is not too large.

Paralleling some of the analysis in [16] for the general modular FTF algorithm, one can show that with the choice of feedback coefficients as in (25), the numerical error propagation in the SWC algorithm is exponentially stable for $\lambda = 1$. This is in sharp contrast with the prewindowed and GWC algorithms, where the absence of exponential weighting leads to random-walk-type numerical errors, irrespective of the feedback coefficients. We had originally expected to find the same phenomenon for the SWC algorithm and had therefore added the exponential weighting factor. Though the LS cost function would then be influenced by two parameters, L and λ, we had expected to determine the (effective) window length mainly via L and to adjust λ within a range very close to unity for optimal numerical performance. However, it can be shown that for a given L, the optimal numerical stability is obtained for $\lambda = 1$. F_{33} (see [18],[16]) is the critical

part in this consideration. With an optimal choice for K_1, the maximal stabilizing effect of the rank-one correction term in F_{33} (see [16]-(42)) can be shown [1],[14] to be approximately $\frac{1}{2N}I$ (for large N). In the SWC algorithm, this maximal stabilization can be achieved irrespective of λ. So we get for the combined effect of the two experiment updates

$$F_{33} \;\approx\; \frac{1}{\lambda}\left(1-\frac{1}{N}\right) I \tag{26}$$

Maximum stability is obviously achieved for $\lambda = 1$. The property of exponential stability for $\lambda = 1$ is actually a great asset of the SWC algorithm, which shows in ill-conditioned cases. Indeed, when we have non-persistent excitation, the rank-one feedback term in F_{33} (or even F_{11}) is inactive in the nullspace of the input covariance matrix. This means that with $\lambda < 1$, there will be exponential error blow-up in this subspace (as in the conventional RLS algorithm [17]). The SWC FTF algorithm, which has a finite memory length even for $\lambda = 1$, will show the much milder random-walk behavior in this subspace (if $\lambda = 1$), just like the robust LMS algorithm.

Even with $\lambda = 1$ though, the proposed feedback mechanism is crucial to obtain a stable numerical behavior in the SWC algorithm. This is illustrated in the following example in which $L = 30$, $N = 10$, $\lambda = 1$, and the input is a white noise signal of unity variance. In Fig. 1, the error signal, $\log\Big(\Big|r^{pf}_{N,L-1}(T) - r^{ps}_{N,L-1}(T)\Big|\Big)$, is plotted as a function of time. The feedback coefficients are chosen as in (25) for the stabilized algorithm, whereas they are $K_1 = 0$, $K_2 = 1$ for the unstable algorithm. Even though $\lambda = 1$, the (surprising) exponential instability of the unstable algorithm can be understood in the same way as the exponential stability of the stabilized algorithm. Summarizing, we may compare the SWC algorithm with the prewindowed algorithm with exponential weighting and conclude that the SWC algorithm is a more numerically robust algorithm for comparable (if not more desirable?) tracking characteristics, at twice the computational cost.

Suitable position for Figure 1

6 Concluding Remarks

The prewindowing assumption leads to the simplest algorithms for fast RLS adaptive filtering, and is therefore widely used. Furthermore, this assumption is a good approximation for the real

data in certain communications applications. From an algorithmic point of view on the other hand, this paper also establishes the fundamental nature of the prewindowing case for RLS problems. Indeed, we have shown how the growing- and sliding-window covariance methods can be embedded into prewindowed problems. Applying the modular prewindowed algorithm then yields the existing covariance algorithms with little extra effort beyond the prewindowing framework. The covariance algorithms though were originally derived using 'scalar" operations (as in the modular approach), but based on more extensive clever exploitation of the specific structure of the covariance problems. This unifying character of modular prewindowed algorithms holds for all RLS-type algorithms, including besides the FTF group also the Fast Lattice RLS algorithms [6],[12],[8],[9],[7] (the embedding of covariance lattice algorithms into prewindowed lattice algorithms was considered in [13] but failed to produce a framework because of the absence of modularity), and even the conventional RLS algorithms, though the issues become fairly trivial there. Actually, modularity is the missing element in the "unified geometric theory" of [8], which was essentially geared towards the prewindowed case and failed to accommodate the covariance cases nicely.

The prewindowed embedding has allowed us to straightforwardly extend the numerical stabilization of prewindowed FTF algorithms to the covariance algorithms and these considerations have revealed the stabilized SWC algorithm to be a numerically very robust FTF algorithm. Also, since the modular approach corresponds to factorized estimation, the underlying triangular factorizations discussed in [16] apply to the covariance algorithms (even in the single-channel case!) in several ways. Finally, we feel that the added performance that covariance algorithms may bring about, in fast start-up problems for the GWC algorithms or in tracking non-stationarities for the SWC algorithms, is often overlooked.

References

[1] A. Benallal and A. Gilloire. "A New Method to Stabilize Fast RLS Algorithms based on a First-Order Model of the Propagation of Numerical Errors". In *Proc. ICASSP 88 Conf.*, pages 1373–1376, New York, April 1988.

[2] J.M. Cioffi. *Fast Transversal Filters for Communications Applications.* PhD thesis, Stanford University, Stanford, CA, March 1984.

[3] 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.

[4] J.M. Cioffi and T. Kailath. "Windowed Fast Transversal Filters Adaptive Algorithms with Normalization". *IEEE Trans. on ASSP*, ASSP-33(3):607–625, June 1985.

[5] S. Haykin. *Adaptive Filter Theory.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[6] D.T.L. Lee, M. Morf, and B. Friedlander. "Recursive least-squares ladder estimation algorithms". *IEEE Trans. ASSP*, ASSP-29(3):627–641, June 1981.

[7] H. Lev-Ari. "Modular Architectures for Adaptive Multichannel Lattice Algorithms". *IEEE Trans. ASSP*, ASSP-35(4):543–552, April 1987.

[8] H. Lev-Ari, T. Kailath, and J. Cioffi. "Least-squares adaptive lattice and transversal filters: a unified geometric theory". *IEEE Trans. Info. Theory*, IT-30(2):222–236, March 1984.

[9] F. Ling and J.G. Proakis. "A Generalized Multichannel Least Squares Lattice Algorithm based on Sequential Processing Stages". *IEEE Trans. ASSP*, ASSP-32(2):381–390, April 1984.

[10] B. Porat. *Contributions to the Theory and Applications of Lattice Filters.* PhD thesis, Stanford University, Stanford, CA, Aug. 1982.

[11] B. Porat. "Second-order equivalence of rectangular and exponential windows in least-squares estimation of Gaussian autoregressive processes". *IEEE Trans. ASSP*, ASSP-33(5):1209–1212, Oct. 1985.

[12] B. Porat, B. Friedlander, and M. Morf. "Square-root covariance ladder algorithms". *IEEE Trans. Automatic Control*, AC-27:813–829, Aug. 1982.

[13] B. Porat, M. Morf, and D.R. Morgan. "On the Relationship Among Several Square-root Normalized Ladder Algorithms". In *Proc. CISS 81*, pages 496–501, Johns Hopkins Univ., Baltimore, MD, March 28 1981.

[14] D.T.M. Slock. "An Overview of Some Recent Advances in Fast RLS Algorithms". In E. Deprettere and A.-J. van der Veen, editors, *Algorithms and Parallel VLSI Architectures*, Elsevier Science Publishers, Amsterdam, 1991. Proc. of Int. Workshop, Pont-à-Mousson, France, June 10-16, 1990.

[15] D.T.M. Slock, L. Chisci, H. Lev-Ari, and T. Kailath. "Modular and Numerically Stable Fast Transversal Filters for Multichannel and Multiexperiment RLS". IEEE Trans. Signal Processing, April 1992.

[16] D.T.M. Slock and T. Kailath. "A Modular Multichannel Multiexperiment Fast Transversal Filter RLS Algorithm". Submitted to Signal Processing.

[17] D.T.M. Slock and T. Kailath. "Numerically Stable Fast Recursive Least-Squares Transversal Filters". In *Proc. ICASSP 88 Conf.*, pages 1365–1368, New York, April 1988.

[18] 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.

List of Footnotes appearing in the Text

Footnote 1: In this paper, superscript H denotes Hermitian (complex conjugate) transpose.

Footnote 2: The Kronecker product $\otimes$ of two matrices is defined as $A \otimes B \stackrel{\triangle}{=} [a_{ij}B]$, a block matrix in which block (i,j) is $a_{ij}B$. The direct sum $\oplus$ of matrices is defined as $\bigoplus_{k=1}^{m} A_k = A_1 \oplus A_2 \oplus \cdots \oplus A_m \stackrel{\triangle}{=} \text{block-diag}\{A_1, A_2, \ldots, A_m\}$.

Table I

Definition of Modular Multichannel Covariance FTF Quantities

Variable	Definition	TF Computation
Kalman gains		
$C_{j,N,L,T}$	$-\sigma^H \Lambda_L K_{j,N,L,T}$	
$\tilde{C}_{j,N,L,T}$	$-X^H_{j,N}(T)\lambda^{-1} R^{-1}_{j,N,L-1,T-1}$	$\gamma^{-1}_{j,N,L}(T)\ C_{j,N,L,T}$
$D_{j,N,L,T}$	$-\pi^H \Lambda_L K_{j,N,L,T}$	
$\tilde{D}_{j,N,L,T}$	$-X^H_{j,N}(T-L+1) R^{-1}_{j,N,L-1,T}$	$\delta^{-1}_{j,N,L}(T)\ D_{j,N,L,T}$
likelihood variables		
$\gamma_{j,N,L}(T)$	$\sigma^H \Lambda_L P^{\perp}_{j,N,L,T} \sigma$	$1 + C_{j,N,L,T} X_{j,N}(T)$
$\gamma^{-1}_{j,N,L}(T)$		$1 - \tilde{C}_{j,N,L,T} X_{j,N}(T)$
$\delta_{j,N,L}(T)$	$\pi^H \Lambda_L P^{\perp}_{j,N,L,T} \pi$	$\lambda^{L-1}\left(1 + D_{j,N,L,T} X_{j,N}(T-L+1)\right)$
$\delta^{-1}_{j,N,L}(T)$		$\lambda^{1-L} - \tilde{D}_{j,N,L,T} X_{j,N}(T-L+1)$
$\eta_{j,N,L}(T)$	$\pi^H \Lambda_L P^{\perp}_{j,N,L,T} \sigma$	$\lambda^{L-1} X^H_{j,N}(T-L+1) C^H_{j,N,L,T}$
		$= D_{j,N,L,T} X_{j,N}(T)$
$\eta^p_{j,N,L-1}(T)$	predicted	$D_{j,N,L-1,T-1} X_{j,N}(T)$
$\eta^s_{j,N,L-1}(T)$	smoothed	$X^H_{j,N}(T-L+1) C^H_{j,N,L-1,T}$
filters		
$A_{j,N,L,T}$	$\left[\ -x^H_{j,L,T} \Lambda_L K_{j-1,N,L,T}\ \ I\right] \mathcal{P}_{j-1}$	
$B_{j,N,L,T}$	$\left[-x^H_{j,L,T-N_j} \Lambda_L K_{j,N,L,T}\ \ I\right] \mathcal{P}_j$	
$W_{N,L,T}$	$-d^H_{L,T} \Lambda_L K_{N,L,T}$	
error covariances		
$\alpha_{j,N,L}(T)$	$x^H_{j,L,T} \Lambda_L P^{\perp}_{j-1,N,L,T} x_{j,L,T}$	$A_{j,N,L,T} X^H_{j,N+1,L,T} \Lambda_L x_{j,L,T}$
$\beta_{j,N,L}(T)$	$x^H_{j,L,T-N_j} \Lambda_L P^{\perp}_{j,N,L,T} x_{j,L,T-N_j}$	$B_{j,N,L,T} X^H_{j,N+1,L,T} \Lambda_L x_{j,L,T-N_j}$
$\xi_{N,L}(T)$	$d^H_{L,T} \Lambda_L P^{\perp}_{N,L,T} d_{L,T}$	$d^H_{L,T} \Lambda_L d_{L,T} + W_{N,L,T} X^H_{N,L,T} \Lambda_L d_{L,T}$
joint-process errors		
$\epsilon_{N,L}(T)$	$d^H_{L,T} \Lambda_L P^{\perp}_{N,L,T} \sigma$	$d(T) + W_{N,L,T} X_N(T)$
$\epsilon^p_{N,L-1}(T)$	predicted	$d(T) + W_{N,L-1,T-1} X_N(T)$
$\nu_{N,L}(T)$	$d^H_{L,T} \Lambda_L P^{\perp}_{N,L,T} \pi$	$\lambda^{L-1}\left(d(T-L+1) + W_{N,L,T} X_N(T-L+1)\right)$
$\nu^s_{N,L-1}(T)$	smoothed	$d(T-L+1) + W_{N,L-1,T} X_N(T-L+1)$

Table II

Modular Multichannel GWC FTF Algorithm

#	**Computation**	**Cost** ($\times$)
Prediction Problem	$L = T - N_m + 2$	
For $j = 1, \ldots, p$ do :		
1	$\left(A_{j,N,L-1,T}, \alpha^{-1}_{j,N,L-1}(T), \tilde{C}_{j,N+1,L-1,T}, \gamma^{-s}_{j,N+1,L-1}(T)\right)$	$(1\div)$
	$= f_U\left(A_{j,N,L-2,T-1}, \lambda^{-1}\alpha^{-1}_{j,N,L-2}(T-1), \left[\tilde{C}_{j-1,N,L-1,T} \quad 0\right]\mathcal{P}_{j-1}, \gamma^{-s}_{j-1,N,L-1}(T), X_{j,N+1}(T)\right)$	$3N+6$
2	$\left(B_{j,N,L-1,T}, \beta_{j,N,L-1}(T), \left[\tilde{C}_{j,N,L-1,T} \quad 0\right]\mathcal{P}_j, \gamma^{-s}_{j,N,L-1}(T)\right)$	$(1\div)$
	$= f_D\left(B_{j,N,L-2,T-1}, \lambda\beta_{j,N,L-2}(T-1), \tilde{C}_{j,N+1,L-1,T}, \gamma^{-s}_{j,N+1,L-1}(T), X_{j,N+1}(T)\right)$	$3N+8$
end do.		
3	$\tilde{C}^{\overline{N}}_{p+1,\overline{N}+1,T} = -\tilde{C}_{p,N,L-1,T}X_N(N_m-1)$	N
4	$\left([D_{N,L,T} \quad 1], \delta_{N,L}(T), \left[\tilde{C}_{N,L,T} \quad 0\right], \gamma^{-s}_{N,L}(T)\right)$ (see section 5.1 also)	$(1\div)$
	$= f_D\left([D_{N,L-1,T-1} \quad 1], \lambda\delta_{N,L-1}(T-1), \left[\tilde{C}_{p,N,L-1,T} \quad \tilde{C}^{\overline{N}}_{p+1,\overline{N}+1,T}\right], \gamma^{-s}_{p,N,L-1}(T), \left[X^H_N(T) \quad 0\right]^H\right)$	$3N+8$
5	$\gamma_{N,L}(T) = \gamma^a_{N,L}(T) = \lambda^{\overline{N}-T-1}\delta_{N,L}(T)\prod_{j=1}^{p}\left(\beta_{j,N,L-1}(T)\alpha^{-1}_{j,N,L-1}(T)\right)$	$2p+4$
	$\gamma^{-s}_{0,N,L}(T+1) = \gamma^{-1}_{N,L}(T)$, $\tilde{C}_{0,N,L,T+1} = \tilde{C}_{N,L,T}$	
Joint-Process Extension		
6	$(W_{N,L,T}, \epsilon_{N,L}(T)) = f_J\left(W_{N,L-1,T-1}, \tilde{C}_{N,L,T}, \gamma_{N,L}(T), d(T), X_N(T)\right)$	$2N+1$
p-channel total cost ($2p+1$ divisions):		$(6p+6)N + 16p + 13$

Table III

Modular Multichannel SWC FTF Algorithm

#	**Computation**	**Cost** ($\times$)
Prediction Problem		
For $j = 1, \ldots, p$ do :		
1	$\left(A_{j,N,L,T}, \alpha^{-1}_{j,N,L}(T), \tilde{C}_{j,N+1,L,T}, \gamma^{-s}_{j,N+1,L}(T)\right)$	$(1\div)$
	$= f_U\left(A_{j,N,L-1,T-1}, \lambda^{-1}\alpha^{-1}_{j,N,L-1}(T-1), \left[\tilde{C}_{j-1,N,L,T} \quad 0\right]\mathcal{P}_{j-1}, \gamma^{-s}_{j-1,N,L}(T), X_{j,N+1}(T)\right)$	$3N+6$
2	$\left(B_{j,N,L,T}, \beta_{j,N,L}(T), \left[\tilde{C}_{j,N,L,T} \quad 0\right]\mathcal{P}_j, \gamma^{-s}_{j,N,L}(T)\right)$	$(1\div)$
	$= f_D\left(B_{j,N,L-1,T-1}, \lambda\beta_{j,N,L-1}(T-1), \tilde{C}_{j,N+1,L,T}, \gamma^{-s}_{j,N+1,L}(T), X_{j,N+1}(T)\right)$	$3N+8$
end do.	$\tilde{C}_{N,L,T} = \tilde{C}_{0,N,L,T+1} = \tilde{C}_{p,N,L,T}$	
For $j = 1, \ldots, p$ do :		
3	$\left(A_{j,N,L-1,T}, \alpha^{-1}_{j,N,L-1}(T), \widehat{D}_{j,N+1,L,T}, -\widehat{\delta}^{s}_{j,N+1,L}(T)\right)$	$(1\div)$
	$= f_U\left(A_{j,N,L,T}, \alpha^{-1}_{j,N,L}(T), \left[\widehat{D}_{j-1,N,L,T} \quad 0\right]\mathcal{P}_{j-1}, -\widehat{\delta}^{s}_{j-1,N,L}(T), X_{j,N+1}(T-L+1)\right)$	$3N+5$
4	$\left(B_{j,N,L-1,T}, \beta_{j,N,L-1}(T), \left[\widehat{D}_{j,N,L,T} \quad 0\right]\mathcal{P}_j, -\widehat{\delta}^{s}_{j,N,L}(T)\right)$	$(1\div)$
	$= f_D\left(B_{j,N,L,T}, \beta_{j,N,L}(T), \widehat{D}_{j,N+1,L,T}, -\widehat{\delta}^{s}_{j,N+1,L}(T), X_{j,N+1}(T-L+1)\right)$	$3N+7$
end do.	$\widehat{D}_{N,L,T} = \widehat{D}_{0,N,L,T+1} = \widehat{D}_{p,N,L,T}$	
5	T even : $\gamma_{N,L}(T) = \gamma^{a}_{N,L}(T) = \lambda^{N+L-1}\widehat{\delta}^{s}_{p,N,L}(T)\prod_{j=1}^{p}\left(\beta_{j,N,L-1}(T)\alpha^{-1}_{j,N,L-1}(T)\right) = \gamma^{s}_{0,N,L}(T+1)$	$2p+3$
	$\widehat{\delta}_{N,L}(T) = \widehat{\delta}^{s}_{p,N,L}(T) = \widehat{\delta}^{s}_{0,N,L}(T+1)$	
6	T odd : $\widehat{\delta}^{-1}_{N,L}(T) = \widehat{\delta}^{-a}_{N,L}(T) = \lambda^{N+L-1}\gamma^{-s}_{p,N,L}(T)\prod_{j=1}^{p}\left(\beta_{j,N,L-1}(T)\alpha^{-1}_{j,N,L-1}(T)\right) = \widehat{\delta}^{-s}_{0,N,L}(T+1)$	$2p+3$
	$\gamma^{-1}_{N,L}(T) = \gamma^{-s}_{p,N,L}(T) = \gamma^{-s}_{0,N,L}(T+1)$	
Joint-Process Extension		
7	$(W_{N,L,T}, \epsilon_{N,L}(T)) = f_J\left(W_{N,L-1,T-1}, \tilde{C}_{N,L,T}, \gamma_{N,L}(T), d(T), X_N(T)\right)$	$2N+1$
8	$(W_{N,L-1,T}, -\lambda^{L-1}\nu^{s}_{N,L-1}(T)) = f_J\left(W_{N,L,T}, \widehat{D}_{N,L,T}, -\widehat{\delta}^{-1}_{N,L}(T), d(T-L+1), X_N(T-L+1)\right)$	$2N+1$
p-channel total cost ($4p$ divisions):		$(12p+4)N + 28p + 5$

List of Figures

Figure 1: