# The Block Underdetermined Covariance (BUC) Fast Transversal Filter (FTF) Algorithm for Adaptive Filtering

Dirk T.M. Slock
Eurecom Institute
2229 route des Crêtes, Sophia Antipolis
F-06560 Valbonne, FRANCE

## Abstract

*We consider the covariance formulation, or hence a rectangular window for the least-squares (LS) criterion. We are specifically interested in the case when the window length $L$ is shorter than the FIR filter order $N$, leading to an underdetermined LS problem. We develop Fast algorithms in Transversal Filter form (FTF) for an adaptive filtering strategy consisting of treating consecutive blocks of $L$ data. The resulting BUC FTF algorithm can be situated in between the Normalized Least-Mean-Square (NLMS) algorithm for $L = 1$ and one extreme case of the Block-processing FTF algorithm for $L = N$. A projection mechanism onto a subspace of dimension $L$ renders their convergence less sensitive to the coloring of the input signal spectrum than is the case for the NLMS algorithm. Their underdetermined LS character additionally endows them with relatively fast tracking characteristics. Relations to existing algorithms and various implementations with varying degrees of complexity are also discussed.*

## 1 Introduction

The tracking characteristics of the (N)LMS algorithm are inherent in the iterative nature of the algorithm; the rate of convergence depends on the eigenvalue spread of the input covariance matrix. For Recursive LS (RLS) algorithms, the tracking is independent of the eigenvalue spread and can be made arbitrarily fast by appropriate choice of the (effective) window length. Fast RLS algorithms and especially the FTF algorithms [1] take only $7N/8N$ (stabilized form) flops with prewindowing (and exponential weighting), or $14N/15N$ flops in the Sliding Window Covariance (SWC) formulation [2],[3]. This should be compared to the $2N$ flops for the LMS or Normalized LMS algorithms.

Here we propose a new class of algorithms which are intermediate between LMS and fast RLS in complexity, and in tracking performance (or sometimes the best). Before we get more specific, let us consider the following motivations.

### 1.1 Stochastic Considerations

A stochastic Newton algorithm is characterized by the following update equations:

$$\begin{aligned} \epsilon^p_k &= d_k + W_{k-1}X_k \\ W_k &= W_{k-1} - \mu\, \epsilon^p_k X^H_k R^{-1} \end{aligned} \qquad (1)$$

where ideally $R = \mathrm{E}\, X_k X^H_k$ is used to compensate for the non-white spectrum of the input signal $x_k$. ($X_k = \left[x^H_k \cdots x^H_{k-N+1}\right]^H$ is the data vector.) The NLMS algorithm omits this decorrelation ($R = \|X_k\|^2 I$), while the RLS algorithm uses the sample covariance matrix $R_k$ as an estimate for $\frac{1}{\mu}R$. One should notice that in the RLS algorithm, the size of the decorrelation matrix is automatically put equal to the length of the (FIR) filter that is being adapted. In many applications however, this coupling may not be well motivated. Consider for instance the problem of acoustic echo cancellation. Here, FIR filters of very great lengths are being used. On the other hand, the input signal is typically a speech signal for which in speech coding, prediction filters with an order of typically only 10 are used (to whiten the signal). So one could consider using a banded matrix for $R^{-1}$ in (1) (corresponding to an AR model for the input) with a (one-sided) bandwidth of only 10. This has been proposed in [4], in which non-fast and fast recursive algorithms are then developed. The resulting algorithm could be considered to be of the symmetric Instrumental Variable type. Note that this approach corresponds very closely to first sending the input signal through an adaptive prediction filter of order 10, and then using the whitened input signal as input to the (N)LMS algorithm. This last approach has been proposed in [5]. One may remark that in this last approach, when considering tracking non-stationarities, the lag in the prewhitening process

1058-6393/92 $03.00 © 1992 IEEE

has to be added to the lag of the (N)LMS algorithm. In the Fast Newton Transversal Filters (FNTF) of [4] on the other hand, some peculiarities have been observed [6] when the input signal is nonstationary, due to the fact that the optimal low-order prediction filter may change considerably over the length of the adapted FIR filter.

### 1.2 Deterministic Considerations

Nevertheless, the FNTF algorithm, which is intermediate between the LMS and RLS algorithms in various ways, yields a significant improvement in convergence speed (in particular a substantial reduction in sensitivity to eigenvalue disparity of $R$) when the input signal is strongly colored. However, it is useful to also take a look at some deterministic aspects. Indeed, based on only the above stochastic considerations, it is not possible to explain why the RLS algorithm converges significantly faster than the (N)LMS algorithm (e.g. by a factor of 5) even in the white input signal case! This advantage of the RLS algorithm is due to the fact that the RLS algorithm solves a (overdetermined) set of equations exactly at each time step ($N$ equations are sufficient to get an unbiased estimate of the $N$ FIR filter coefficients). Although asymptotically for large window lengths, the rectangular window and the exponential window give equivalent performance for corresponding window lengths [7], there is reason to believe that the rectangular window leads to better tracking performance for short windows in strongly non-stationary environments.

The Block-Processing RLS algorithm [8] provides the least-squares solution to $L \geq N$ equations, which are obtained by writing out the error signal at $L$ consecutive time steps:

$$\min_{W_k}\left\{\|d_{L,k} - X_{N,L,k}W_k^H\|^2\right\} \qquad (2)$$

where

$$\begin{aligned} d_{L,k} &= [d_k \cdots d_{k-L+1}]^H \\ X_{N,L,k} &= \begin{bmatrix} x_k^H & \cdots & x_{k-N+1}^H \\ \vdots & & \vdots \\ x_{k-L+1}^H & \cdots & x_{k-N-L+2} \end{bmatrix} . \end{aligned} \qquad (3)$$

When $L < N$, the above problem is underdetermined. A unique solution can still be found though by taking the minimum-norm solution. In fact, the NLMS algorithm (with normalized stepsize $\overline{\mu} = 1$) provides this minimum-norm solution for $L = 1$. We shall consider algorithms for the cost function (2) with $1 \leq L \leq N$, which hence cover a continuous range between the Block-Processing RLS algorithm and the NLMS algorithm, and which have the deterministic projection interpretation of making $L$ consecutive error samples exactly zero.

## 2 A Unified Least-Squares Criterion

We shall mostly adhere to the notation of [1],[2],[3]. The following criterion will prove to encompass quite a number of different adaptation techniques. Consider

$$\min_{W_{N,k}}\left\{\|d_{L,k}-X_{N,L,k}W_{N,k}^H\|^2_{S_k^{-1}}+\|W_{N,k}^H-W_{N,k-M}^H\|^2_{T_k^{-1}}\right\} \qquad (4)$$

where $\|v\|_S^2 = v^H S v$ and $S_k$, $T_k$ are Hermitian positive definite matrices. Before discussing various applications of this formulation, let us consider the minimization of this criterion (4). Setting the gradient equal to zero, and applying the Matrix Inversion Lemma (MIL) to solve the resulting linear equation in $W_{N,k}$, we get

$$\begin{aligned} W_{N,k} &= W_{N,k-M} + \left(d_{L,k}^H - W_{N,k-M}X_{N,L,k}^H\right) \\ &\quad \times \left(X_{N,L,k}T_kX_{N,L,k}^H + S_k\right)^{-1} X_{N,L,k}T_k \ . \end{aligned} \qquad (5)$$

The Hessian of the quadratic criterion (4) is

$$H_k = T_k^{-1} + X_{N,L,k}^H S_k^{-1} X_{N,L,k} \qquad (6)$$

to which we have applied the MIL to obtain (5). Usually, $M = 1$, unless we consider block processing. In all applications we consider here, the idea is to equalize the nature of both terms that constitute $H_k$. In a first class of algorithms, we take $L = 1$ and $S_k = I$, which renders the nature of the second term in (6) of a covariance type (be it of low rank). Hence, $T_k^{-1}$ is chosen to be representative of the covariance matrix of $X_N(k)$. In the stochastic Newton algorithms, we choose (apart from a scalar multiple) $T_k^{-1} = R_N = \text{E}\ X_N(k)X_N^H(k)$. In the absence of second order statistics, the RLS algorithms with exponential weighting factor $\lambda$ uses $T_k^{-1} = \lambda R_{N,k-1}$ where $R_{N,k} = \sum_{i=0}^k \lambda^{k-i} X_N(i)X_N^H(i)$. In the Fast Newton algorithm [4], $R_{N,k}$ is replaced by the Maximum Entropy extension of a banded restriction of $R_{N,k}$ (with banded inverse).

In a second class of algorithms, one takes $T_k = I$, which has all eigenvalues equal to a constant. Hence, now $S_k$ should be chosen such that the eigenvalues of the second term in (6) are either constant or zero (since the term is not of full rank). The way to achieve this exactly is to make the second term in (6) proportional to a projection matrix and hence to choose $S_k = \mu_k X_{N,L,k}X_{N,L,k}^H$. With $L < N$, we have an "underdetermined" problem (strictly speaking, we only get an underdetermined problem as $\mu_k \to 0$). The filter solution (5) can now be rewritten as

$$W_{N,k} = W_{N,k-M} + \frac{1}{1+\mu_k}\left(d^H_{L,k} - W_{N,k-M}X^H_{N,L,k}\right) R^{-1}_{L,N,k}X_{N,L,k} \tag{7}$$

where $R_{L,N,k} = X_{L,N,k}X^H_{L,N,k}$ is the sample covariance matrix for a Block-Processing problem with filter length $L$ and window length $N$ interchanged. One immediately recognizes the NLMS algorithm when $M = L = 1$. Using (7), one can establish the following relation between the *a posteriori* and the *a priori* error vectors

$$d^H_{L,k} - W_{N,k}X^H_{N,L,k} = \frac{\mu_k}{1+\mu_k}\left(d^H_{L,k} - W_{N,k-M}X^H_{N,L,k}\right) . \tag{8}$$

One may note that the convergence of (7) is governed by a product of matrices of the form $I - \frac{1}{1+\mu_k}P_{X^H_{N,L,k}}$ (where $P_X$ is the projection matrix onto the column space of $X$), from which one can see a prewhitening effect of order $L{-}1$ transpire. In particular for $\mu_k = 0$, this matrix becomes the projection matrix $P^{\perp}_{X^H_{N,L,k}}$, leading to $L$ a posteriori errors being zeroed.

## 3 The BUC FTF Algorithm

In the BUC FTF algorithm, we process consecutive blocks of $L$ data. The algorithm is the one described by update equation (7) with $M = L$. We can split the update operation into the following steps:

1. compute $L$ a priori filtering errors ($NL$ multiplications):
$$\epsilon_{L,k} = d_{L,k} - X_{N,L,k}W^H_{N,k-L} \tag{9}$$
2. update $R_{L,N,k}$ from $R_{L,N,k-L}$ (close-to-Toeplitz matrix) ($O(L\log L)$ multiplications)
3. use the *generalized Levinson* algorithm to solve for $\eta_{L,k}$ from ($5.5L^2 + o(L^2)$ multiplications)
$$R_{L,N,k}\eta_{L,k} = \frac{1}{1+\mu_k}\epsilon_{L,k} \tag{10}$$
4. update the filter estimate according to ($NL$ multiplications)
$$W_{N,k} = W_{N,k-L} + \eta^H_{L,k}X_{N<L,k} \tag{11}$$

As far as step 2 is concerned, the generalized Levinson algorithm actually requires only the first row of $R_{L,N,k}$. To update this row in a computationally efficient manner, it is desirable that $N$ is an integer multiple of $L$. In that case, the innner products over a time span of $N$ samples can be regarded as a sum of $\frac{N}{L}$ inner products over a time span of $L$ samples. In that way, a length $N$ inner product can be computed recursively from $k-L$ to $k$ by adding in the newest length $L$ inner product and subtracting out the oldest one. The calculation of the $L$ new length $L$ inner products can be done efficiently using FFT techniques which we shall elaborate upon further below, leading to $O(L\log L)$ operations.

In step 3, the close-to-Toeplitz set of equations can be solved efficiently using the generalized Levsinson algorithm. This algorithm is spelled out in Table IV of [8]. The "prediction part" of the algorithm (generating the backward prediction filters $B_{L,N,k}$ and error variances $\beta_{L,N,k}$) can be kept completely, but the "joint-process" part in equations 3,4 should be replaced (due to the different right hand side in our set of equations) by

$$\eta_{l,k} = \begin{bmatrix} \eta_{l-1,k} \\ 0 \end{bmatrix} + \frac{1}{1+\mu_k} B^H_{l-1,N,k}\beta^{-1}_{l-1,N,k}B_{l-1,N,k}\epsilon_{l,k} \tag{12}$$

for $l = 1,\ldots,L$. This small modification does not change the computations count indicated in Table IV of [8], which has $5.5L^2$ as its most significant term. The significant terms in the overall complexity are thus $2NL + 5.5L^2$ for treating $L$ samples, or hence $2N + 5.5L$ per sample. This should be compared with $2N$ for the LMS algorithm.

### 3.1 Superfast BUC FTF Algorithm

Just as the LMS algorithm complexity has recently been shown to be susceptible to further reductions [9], we can also reduce the $2N$ term in the computational complexity of the BUC FTF algorithm. This is done by using FFT techniques for the filtering of blocks of data. Typically, we need to consider a product of the form $X\,v$ where $X$ is a $L \times L$ Hankel data matrix and $v$ is a $L \times 1$ vector. We can embed $X$ into a circulant matrix $\overline{X}$ of dimension $2L$

$$\overline{X} = \begin{bmatrix} X & * \\ * & * \end{bmatrix} \tag{13}$$

We also pad $v$ with $L$ zeros to get

$$\begin{aligned} X\,v &= [I\ \ O]\,\overline{X}\begin{bmatrix} v \\ 0 \end{bmatrix} \\ &= [I\ \ O]\,\overline{X}\,W^H W\begin{bmatrix} v \\ 0 \end{bmatrix} \\ &= [I\ \ O]\,W^H D\, W\begin{bmatrix} v \\ 0 \end{bmatrix} \end{aligned} \tag{14}$$

where $W$ is the unitary $2L$ point DFT matrix, and $D$ is a diagonal matrix with on the diagonal the DFT of the first row of $\overline{X}$ (we exploited the fact that a circulant matrix has an eigen decomposition of the form $\overline{X} = W^H D W$). What (14) says is that we can obtain the product $Xv$ by taking the DFT of $v$ padded with zeros, taking the DFT of the first row of $\overline{X}$, multiplying the two, and keeping the first $L$ elements in the inverse DFT of the product. This takes three $2L$ point DFT's and $2L$ multiplications or hence $\frac{3}{2}L\log(2L) + 2L$ multiplications.

We shall exploit the above strategy by (assuming $\frac{N}{L}$ is an integer) cutting $X_{N,L,k}$ into $\frac{N}{L}$ blocks of $L \times L$ Hankel matrices. The computation of $X_{N,L,k}W^H_{N,k-L}$ can then be carried out by also cutting $W_{N,k-L}$ into $\frac{N}{L}$ segments of length $L$. The update of the filter estimate $W_{N,k}$ is also carried out by dividing the work over the $\frac{N}{L}$ segments. This approach leads to the following dominating terms in the computational complexity per sample

$$2N\frac{\log L}{L} + 5.5L + 4\frac{N}{L} + 3\log L \tag{15}$$

which is easily lower than $2N$. We should note that, as is typical in a block-processing approach, the processing necessitates some delay. This delay can be limited to about $L/2$ samples.

## 4 BUC FTF Convergence Analysis

We shall limit ourselves here to a brief discussion of a number of charactersitics of the BUC FTF algorithm. When the input signal to the adaptive filter has a covariance matrix of rank $L$ (*e.g.* a mixture of $L$ complex exponentials), and consider the system identification setup with no measurement noise, then the BUC FTF algorithm converges exactly in one processing step of $L$ samples! This property is due to the exact LS criterion that the algorithm minimizes. This result should be contrasted with the FNTF algorithm, in which the prewhitening of order $L-1$ is perfectly mached to an AR($L-1$) process as input.

The system dynamics governing the algorithm's convergence are determined by

$$\Phi_k = I - \frac{1}{1+\mu} X^H_{N,L,k} R^{-1}_{L,N,k} X_{N,L,k} \tag{16}$$

For small stepsizes (large $\mu$), we can limit the investigation of the dynamics to an investigation of the mean $\mathrm{E}\,\Phi_k$ (averaging analysis). To this end, when $\frac{N}{L} \gg 1$, we can introduce the following approximation

$$\begin{aligned} &\mathrm{E}\, X^H_{N,L,k} R^{-1}_{L,N,k} X_{N,L,k} \\ &\approx \mathrm{E}\left\{X^H_{N,L,k} \left(\mathrm{E}\, R_{L,N,k}\right)^{-1} X_{N,L,k}\right\} \end{aligned} \tag{17}$$

leading to

$$\mathrm{E}\,\Phi_k = I - \frac{1}{1+\mu}\sum_{l=0}^{L-1} \sigma_l^{-2} R_N^{(l)} \tag{18}$$

where $\sigma_l^2 = R_N^{(l)}(1,1)$ and $R_N^{(l)}$ is the $N \times N$ covariance matrix of the prediction errors of order $l$ of the input signal. $R^{(l)}$ gets more and more diagonally dominant as $l$ increases, and so does $\mathrm{E}\,\Phi_k$. So the influence of the eigenvalue spread of $R_N = R_N^{(0)}$ gets diminished as $l$ increases.

Consider now the system identification setup: the desired response $d_k = W^o X_k + n_k$ is the sum of the output of an optimal filter $W^o$ plus some independent zero-mean i.i.d. measurement noise $n_k$ with variance $\xi^o = \mathrm{E}\, n_k^2$. So the filter estimation error $\widetilde{W}_k = W^o - W_k$ satisfies the following system

$$\begin{aligned} \epsilon_{L,k} &= X_{N,L,k}\widetilde{W}^H_{k-L} + n_{L,k} \\ \widetilde{W}_k &= \widetilde{W}_{k-1}\Phi_k - \tfrac{1}{1+\mu} n^H_{L,k} R^{-1}_{L,N,k} X_{N,L,k}\,. \end{aligned} \tag{19}$$

With $COV_k = \mathrm{E}\left(\widetilde{W}_k^H \widetilde{W}_k\right)$ and introducing the independence assumption (treating $X_{N,L,k}$ and $\widetilde{W}_{k-L}$ as independent), the learning curve becomes

$$\begin{aligned} \xi_k &= \mathrm{E}\,(\epsilon_k)^2 = \mathrm{trace}\left(R_N\, COV_{k-L}\right) + \xi^o \\ COV_k &= \mathrm{E}\,\left(\Phi_k\, COV_{k-L}\, \Phi_k\right) \\ &\quad + \tfrac{\xi^o}{(1+\mu)^2}\, \mathrm{E}\, X^H_{N,L,k} R^{-2}_{L,N,k} X_{N,L,k} \end{aligned} \tag{20}$$

Let us concentrate on the driving term for the $COV_k$ system, and omit the expectation operator. If we have the singular value decomposition $X^H_{N,L,k} = U\Sigma V^H$, then we get

$$X^H\left(XX^H\right)^{-2}X = U\Sigma^{-2}U^H \tag{21}$$

Basically, when the sample covariance matrix $R_{L,N,k}$ gets badly conditioned, then the driving measurement noise term gets amplified significantly. So, the well-known effect in the NLMS algorithm (BUC FTF for $L = 1$), in which the stepsize normalization leads to a potentially higher steady-state MSE (than in the LMS algorithm) due to a bit of noise amplification, this effect gets amplified here since now the conditioning of a covariance matrix of size $L$ comes into play. One remedy would be to regularize $R_{L,N,k}$ in the update equation for $W_{N,k}$. One simple regularization consists of replacing $R_{L,N,k}$ by $R_{L,N,k} + \nu I$ for some small $\nu > 0$. The initialization of the generalized Levinson algorithm allows for a very straightforward incorporation of such a regularization as explained in [8].

When one demands very high tracking capacity, the prewindowed FTF algorithm with exponential weighting is not satisfactory (numerical stability considerations impose a lower bound on the weighting factor). One has to resort to the SWC FTF algorithm, which is also limited in flexiblity however, due to the rectangular window of constant length. The BUC FTF algorithm offers more flexibility: $\mu_k$ can be taken to be time-varying (this is not the same as throwing in a stepsize factor in the SWC algorithm to boost the gain, which makes the algorithm lose its LS nature), $L$ can be varied in time and one can restart at any desired time.

An alternative to an algorithm with $\mu > 0$ is to use the algorithm with $\mu = 0$ and use coefficient filtering [10]. The algorithms with $\mu = 0$ provide the fastest tracking characteristics, leading to a substantial estimation noise component in the excess MSE. This can be reduced with a possibly time-varying trade-off between tracking speed and noise averaging.

In order to get a feeling for the potential speedup that the algorithm can deliver, we have simulated the BUC FTF algorithm on a AR(2) process of which the poles have a radius 0.95 and lie at the angles $\pm 45°$. The filter order is $N = 20$. We compare the algorithm with $L = 1$ (LMS - dashed line) to the algorithm with $L = 3$ (solid line). For comparison purposes, we also ran a simulation of NLMS with white noise as input, to delimit a certain "upper bound" in the convergence speed of the BUC FTF algorithm, which would be achieved if the algorithm could be insensitive to the eigenvalue disparity of $R_N$ (but it isn't, see (18)). Nevertheless, with a relatively low value of $L = 3$, the algorithm is able to converge substantially faster than NLMS.

In [12], we have explored fast transversal algorithms for underdetermined growing and sliding window covariance LS problems. Further connections to other work can be found there also. Basically, in [11], the Block Orthogonal Projection Algorithm (BOPA) was proposed, which minimizes the BUC criterion (with $\mu = 0$). This is a non-fast algorithm (with complexity much higher than that of LMS).

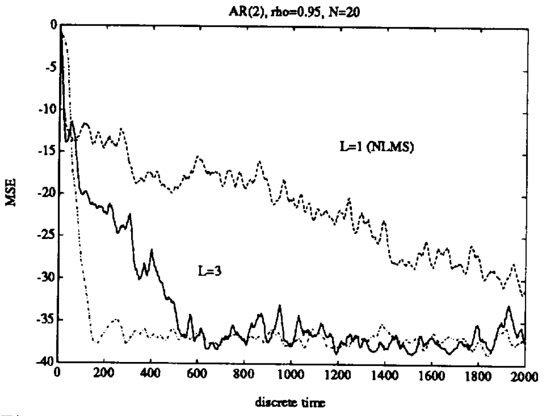

Figure 1: Simulation of the BUC FTF algorithm on an AR(2) process.

## 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] 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.

[3] D.T.M. Slock and T. Kailath. "A Modular Prewindowing Framework for Covariance FTF RLS Algorithms". *Signal Processing*, 28(1):47-61, July 1992.

[4] G.V. Moustakides and S. Theodoridis. "Fast Newton Transversal Filters – A New Class of Adaptive Estimation Algorithms". *IEEE Trans. SP*, SP-39(10):2184-2193, Oct. 1991.

[5] S.H. Leung, B.L. Chang, and S.M. Lau. "On Improving the Convergence Rate of the LMS Algorithm Using Prefiltering". In *Proc. ISSPA90, Signal Processing, Theories, Implementations and Applications*, Gold Coast, Australia, Aug. 27-31 1990.

[6] T. Petillon, A. Gilloire, and S. Theodoridis. "Complexity Reduction in Fast RLS Transversal Adaptive Filters with Application to Acoustic Echo Cancellation". In *Proc. IEEE Int. Conf. ASSP*, San Fransisco, CA, March 1992.

[7] 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.

[8] J.M. Cioffi. "The Block-Processing FTF Adaptive Algorithm". *IEEE Trans. on ASSP*, ASSP-34(1):77-90, Feb. 1986.

[9] J. Benesty and P. Duhamel. "A Fast Exact Least Mean Square Adaptive Algorithm". *IEEE Trans. Sig. Proc.*, SP-40, Dec. 1992.

[10] M. Niedzwiecki. "Identification of Time-Varying Systems using Combined Parameter Estimation and Filtering". *IEEE Trans. ASSP*, 38(4):679-686, April 1990.

[11] T. Furukawa, H. Kubota, and S. Tsuji. "The Orthogonal Projection Algorithm for Block Adaptive Signal Processing". In *Proc. ICASSP 89 Conf.*, pages 1059-1062, Glasgow, Scotland, May 23-26 1989.

[12] D.T.M. Slock. "Underdetermined Growing and Sliding Window Covariance Fast Transversal Filter RLS Algorithms". In *Proc. EUSIPCO 92, VIth European Signal Processing Conference*, pages 1169-1172, Brussels, Belgium, Aug. 24-27 1992.