Joint Time-Delay Estimation and Adaptive Recursive Least Squares Filtering

Daniel Boudreau and Peter Kabal, *Member, IEEE*

Abstract—**A general estimation model is defined in which two observations are available; one being a noisy version of the transmitted signal, while the other is a noisy-filtered and delayed version of the same transmitted signal. The delay and the filter are unknown quantities that must be estimated. An adaptive system, based on the least squares (LS) estimation criterion, is proposed in order to perform a joint estimation of the two unknowns. The joint estimator is conceptually composed of an adaptive delay element operating in conjunction with an adaptive transversal filter. The weighted sum of squared errors is minimized with respect to both the delay and the adaptive filter weight vector. The filter is adapted using a fast version of the recursive least squares (RLS) algorithm, while the delay is updated using a form of derivative, with respect to the delay, of the sum of squared errors. In order to perform this task efficiently, the adaptive delay is limited to integer values and is corrected one sample at a time. The integer delay value is defined as the lag. A series of relations is presented, in order to compute and update the lag value such that the optimum least squares solution is attained. The joint delay estimation and RLS filtering algorithm is obtained by combining the lag update relations with a version of the fast transversal filter RLS algorithm. The simulations of the resulting algorithm show that both stationary and time-varying delays are effectively tracked and that the adaptive filter properly estimates the reference filter impulse response.**

I. Introduction

THE PROBLEM of estimating the time delay between two continuous-time versions of the *same* signal, each one corrupted by uncorrelated noise components, has been the subject of many research efforts in recent years. The conventional continuous-time signal model is of the form

$$z_1(t) = s(t) + v_1(t)$$
$$z_2(t) = As[t - D(t)] + v_2(t), \tag{1}$$

where $s(t)$ is the transmitted signal; $D(t)$ is a time delay, possibly time-varying; A is a constant gain factor; $v_1(t)$ and $v_2(t)$ are zero-mean stationary noise processes, assumed uncorrelated with each other as well as with $s(t)$; while $z_1(t)$ and $z_2(t)$ are the two observed signals. The maximum likelihood estimator for the unknown delay $D(t)$ was derived, for a static [1], [2] and a time-varying delay [3]. Closed-loop adaptive techniques using the minimum mean squared error (MMSE) or the least squares (LS) criteria have also been proposed. In these cases, the estimator structure is such that one signal is processed by an adaptive system, for which the output is compared to the other signal and the error used to adapt a conventional adaptive transversal filter or an adaptive delay element.

In this paper, we consider a signal model that generalizes somewhat the model of (1) by allowing frequency-dependent attenuation in the delayed path. We also specifically consider discrete-time signals and systems. The corresponding model is of the form

$$z_1(n) = s(n) + v_1(n)$$
$$z_2(n) = \mathcal{L}_{D_n,h(n)}[s(n)] + v_2(n) \tag{2}$$

where n is now the discrete-time index; and $\mathcal{L}_{D_n,h(n)}[\cdot]$ is an unknown linear operator taking the form of a filtering operation, with the filter impulse response $h(n)$, of a delayed by D_n version of the signal $s(n)$. A block diagram corresponding to the mathematical model of (2) is illustrated in Fig. 1.

The signal $s(n)$ can be delayed before or after it is filtered. In this paper, we consider the later case, where the operator $\mathcal{L}[\cdot]$ corresponds to a filtering operation of $s(n)$, followed by a discrete-time delay D_n. The mathematical expression for that kind of operation is better expressed using the continuous-time versions of the filter and the input signal. The "filter and delay" operation is then given as a resampling of the filtered version of the continuous-time signal $s(t)$, i.e.

$$\mathcal{L}_{D_n,h(n)}[s(n)] = h(t) \otimes s(t)|_{t=nT-D_n} \tag{3}$$

where the operator $\otimes$ is the convolution operator. The corresponding discrete-time block diagram is illustrated in Fig. 2.

Another form of $\mathcal{L}_{D_n,h(n)}[\cdot]$ corresponds to the filtering of a delayed version of $s(n)$, i.e.,

$$\mathcal{L}_{D_n,h(n)}[s(n)] = h(n) \otimes s(nT - D_n). \tag{4}$$

Note that the time-varying reference delay D_n is not limited to an integer number of sampling periods and can take any real value. All the discrete-time signals are assumed to be the sampled versions, with sampling period T, of continuous-time signals that are strictly bandlimited to the frequency range $-1/2T < f < 1/2T$. Examples of such systems are encountered in system modeling problems,

Manuscript received April 21, 1991; revised January 29, 1992. This work was supported in part by the Canadian Department of Communications and the Natural Science and Engineering Research Council of Canada.

D. Boudreau is with the Communications Research Centre, Directorate of Satellite Communications, Ottawa, Ontario, Canada, K2H 8S2.

P. Kabal is with the Department of Electrical Engineering, McGill University, Montréal, Québec, H3A 2A7, and with INRS-Télécommunications, Université du Québec, Verdun, Québec, Canada, H3E 1H6.

IEEE Log Number 9205106.

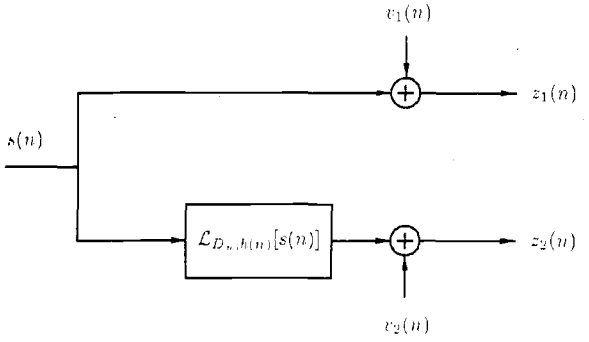


Fig. 1. Mathematical signal model.

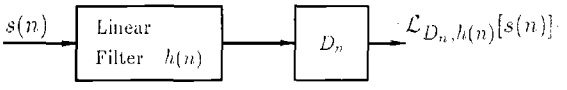


Fig. 2. System model of interest.

where the unknown system often has an impulse response that can be modeled as a pure time delay in series with a linear filter. This can occur in noise or echo cancellation, digital communication, or geophysical exploration.

In order to estimate $\mathcal{L}_{D_n,h(n)}[\cdot]$ or its inverse, we propose to use a joint estimator that is composed, at least conceptually, of an *adaptive delay element* d_n and a conventional Mth-order adaptive FIR filter with weight vector $\mathbf{w}_M(n)$. The joint adaptation is based on the use of the same error signal for the adaptation of both systems, using the least squares criterion. It can be used in system identification (cancellation) mode, in which an error signal is formed by filtering $z_1(n)$ with an estimate of $\mathcal{L}_{D_n,h(n)}[\cdot]$ and subtracting it from $z_2(n)$. It may also be configured in inverse filtering (equalization) mode, in which the error signal results from passing $z_2(n)$ through an estimate of $\mathcal{L}_{-D_n,h^{-1}(n)}[\cdot]$ and comparing with $z_1(n)$. A system identification configuration may take, for example, the form shown in Fig. 3. This configuration is the one that is explicitly assumed throughout the article.

This implies that the reference system that is being estimated corresponds to (3), or that it can be closely approximated as such [e.g., if the reference delay varies slowly in a configuration represented by (4)]. This form of joint estimator may be used in applications where both the reference delay and filter must be estimated. A simple adaptive filter has the potential to model the functional $\mathcal{L}_{D_n,h(n)}[s(n)]$ since this function can be approximated by an FIR filter with the proper number of taps. This approach is inefficient in the sense that the reference delay D_n is modeled by a time shift in the adaptive filter impulse response. For a fixed filter order M, this shift may result in an error that is larger than the error corresponding to perfect modeling. An additional adaptive delay estimation algorithm, specifically designed to track the reference delay variations, allows a better impulse response centering and the use of an adaptive filter with a smaller order.

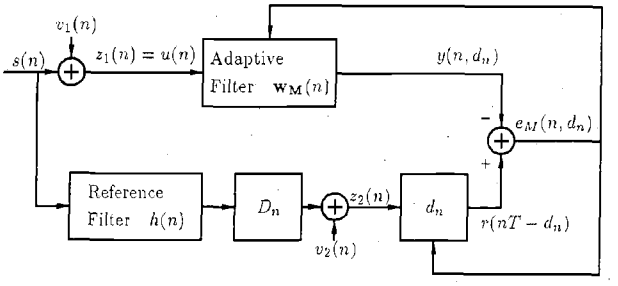


Fig. 3. System identification by a joint adaptive estimator.

The choice of an adaptive least squares filter, under the form of the recursive least squares (RLS) algorithm, is motivated by the fast convergence properties of this algorithm. The simplest form of this algorithm is generated by using the matrix inversion lemma, which gives a computationally involved algorithm [4]. A drastic cut in the number of computations is obtained with the so-called *fast* implementation of the RLS algorithm. Carayannis *et al.* [5] derived the *fast a posteriori error sequential technique* (FAEST), while Cioffi and Kailath [6] worked on a very similar implementation, the *fast transversal filter* (FTF). We develop our joint time delay estimation and adaptive LS filtering algorithm by making use of the FTF[1] form of the RLS algorithm. The reason for this choice is not only the reduced computational complexity, but also the fact that the delay estimation part of the joint algorithm can be naturally linked with the variables used in the FTF algorithm. A major drawback of the FTF algorithm is its possible numerical instability in tracking conditions [6]. Several methods have been proposed to solve this problem, and we rely on a fairly simple restart procedure to circumvent it.

Note that the algorithms discussed in this article are suited for *tracking* the reference system, once a form of acquisition has been performed. The modeling capabilities of the adaptive filter is such that, in general, the mean squared error (and the sum of squared errors) is not unimodal with respect to the delay. Without an acquisition algorithm, it is very likely that the delay estimation algorithm tracks a local minimum. The acquisition problem is briefly addressed in Section III.

The paper is organized as follows. Some notation and background material is presented in Section II. Then the joint RLS algorithm is given in Section III, which is followed by some experimental results in Section IV.

II. Background Theory

The variables encountered in the joint algorithm are defined in the next subsection. Some recursive relations allowing us to update the adaptive delay are given in Section II-B.

[1]FTF is used thereafter as a generic term that includes both the FAEST algorithm and the original FTF algorithm of [6].

A. Some Notation and Definitions

In the *prewindowed weighted recursive least squares* adaptation algorithm for adaptive transversal filters of order M, the index of performance to be minimized, at iteration n, and for an *integer* adaptive delay l[2] in the reference data, is

$$\mathcal{E}(n) = \sum_{i=1}^{n} \beta^{n-i} |e_M(i, l)|^2 \tag{5}$$

where the *a posteriori* estimation error is defined by

$$e_M(i, l) = r(i + l) - \mathbf{w}_M^{lH}(n)\mathbf{u}_M(i) \tag{6}$$

with

$$\mathbf{u}_M(i) = [u(i), u(i-1), \cdots, u(i-M+1)]^T$$

$$\mathbf{w}_M^l(n) = [w_{1M}^l(n), w_{2M}^l(n, \cdots, w_{MM}^l(n)]^T \tag{7}$$

and the superscript H denotes complex conjugate transpose. The sample $u(n)$ is the input to the adaptive filter, and $r(n + l)$ represents the delayed reference signal. The vector $\mathbf{w}_M^l(n)$ is the adaptive filter weight vector used when the lag is l. Note that the prewindowed method assumes that the data is zero prior to iteration $n = 1$ [4].

The constant β is a weighting factor close to, but less than one [4]. Strictly speaking, an algorithm based on (5) is not completely suitable for tracking nonstationary reference signals, since it never completely "forgets" the past data. But for β lower than one, the tracking capabilities are generally acceptable [6].

The least squares solution is obtained as a function of the deterministic autocorrelation matrix, defined as (using the notation in [4])

$$\mathbf{\Phi}_M(n) = \sum_{i=1}^{n} \beta^{n-i} \mathbf{u}_M(i)\mathbf{u}_M^H(i) \tag{8}$$

and of the deterministic cross-correlation vector with lag l given by

$$\boldsymbol{\theta}_M^l(n) = \sum_{i=1}^{n} \beta^{n-i}\, \mathbf{u}_M(i) r^*(i + l). \tag{9}$$

The least squares weight vector at iteration n, for lag l, is

$$\hat{\mathbf{w}}_M^l(n) = \mathbf{\Phi}_M^{-1}(n)\boldsymbol{\theta}_M^l(n) \tag{10}$$

and the corresponding minimum of squared errors is

$$\hat{\xi}_M(n, l) = \min_{\mathbf{w}} \mathcal{E}(n)$$

$$= \sum_{i=1}^{n} \beta^{n-i} |r(i + l) - \hat{\mathbf{w}}_M^{lH}(n)\mathbf{u}_M(i)|^2. \tag{11}$$

Note that the data is assumed to yield a deterministic autocorrelation matrix that is nonsingular.

[2]This integer delay is defined as the lag and is negative. Note that l does not carry a time index because, in the RLS algorithm, it is assumed that the signals are stationary within the memory of the algorithm (defined by β), which implies that l applies to all the previous data.

The FTF form of the RLS algorithm is expressed in terms of four transversal filters [6] that are applied on the input signal. With infinite precision arithmetic, it implements exactly the RLS recursions, with large computational savings. The particularity of the RLS adaptive filter algorithm is that it computes the true solution of the LS problem at each iteration, which typically ensures a rate of convergence one order of magnitude faster than the simple steepest-descent or LMS algorithms [4]. A one-step-forward linear predictor and a one-step-backward linear predictor are first used. They essentially whiten the input signal, and their outputs and impulse responses are used to update a third filter, the Kalman gain filter. The output and weight vector of this third filter are used to update the actual adaptive filter weight vector $\hat{\mathbf{w}}_M^l(n)$. In order to obtain the FTF algorithm, a few more quantities must be defined as follows.

First, the *a priori* estimation error $\alpha_M(i, l)$ is defined as

$$\alpha_M(i, l) = r(i + l) - \mathbf{w}_M^{lH}(n-1)\mathbf{u}_M(i). \tag{12}$$

The optimum weight vector for the one-step forward linear predictor of order m is denoted as $\mathbf{a}_m(n)$ $(1 \le m \le M)$. This vector minimizes the sum of weighted forward *a posteriori* prediction-error squares, defined as

$$F_m(n) = \sum_{i=1}^{n} \beta^{n-i} |f_m(i)|^2 \tag{13}$$

where

$$f_m(i) = u(i) - \mathbf{a}_m^H(n)\mathbf{u}_m(i-1). \tag{14}$$

Then the forward *a priori* prediction-error $\eta_m(i)$ is

$$\eta_m(i) = u(i) - \mathbf{a}_m^H(n-1)\mathbf{u}_m(i-1). \tag{15}$$

Similarly, the optimum weight vector for the one-step backward linear predictor of order m is the vector $\mathbf{b}_m(n)$ that minimizes the sum of weighted backward *a posteriori* prediction-error squares, expressed as

$$B_m(n) = \sum_{i=1}^{n} \beta^{n-i} |b_m(i)|^2 \tag{16}$$

with

$$b_m(i) = u(i-m) - \mathbf{b}_m^H(n)\mathbf{u}_m(i). \tag{17}$$

Then the backward *a priori* prediction-error $\psi_m(i)$ is

$$\psi_m(i) = u(i-m) - \mathbf{b}_m^H(n-1)\mathbf{u}_m(i). \tag{18}$$

We define four more quantities that are used specifically in the lag-recursive relations and the FTF algorithm. The inner-product of the forward *a posteriori* prediction error with the delayed reference signal is given by

$$v_m^{fl}(n) = \sum_{i=1}^{n} \beta^{n-i} f_m(i) r^*(i + l). \tag{19}$$

Similarly, the inner-product of the backward *a posteriori* prediction error with the delayed reference signal is

$$v_m^{bl}(n) = \sum_{i=1}^{n} \beta^{n-i}\, b_m(i) r^*(i + l). \tag{20}$$

The Kalman gain vector $g_M(n)$, as used also in the RLS algorithm, is defined as

$$g_M(n) = \Phi_M^{-1}(n-1)u_M(n) \tag{21}$$

and the "Kalman filter error" $\gamma_M(n)$ as

$$\gamma_M(n) = 1 + \beta^{-1} g_M^H(n)u_M(n). \tag{22}$$

B. The Lag-Recursive Relations

The key point in the joint adaptive algorithm considered in this paper is the availability of a set of lag-recursive relations, that allows us to compute the optimum weight vector and the sum of squared errors for a certain lag, given the same quantities for another lag value. These relations, first derived for block LS estimation by Kalouptsidis *et al.* [7] and obtained using a geometrical framework in [8], are the following:[3]

1) Forward recursions for the error:

$$\hat{\xi}_{M-1}(n-1, l+1) = \hat{\xi}_M(n, l) + \frac{|v_{M-1}^{fl}(n)|^2}{F_{M-1}(n)} \tag{23}$$

$$\hat{\xi}_{M-1}(n, l+1) = \beta\hat{\xi}_{M-1}(n-1, l+1) + \alpha_{M-1}^*(n, l+1)e_{M-1}(n, l+1) \tag{24}$$

$$\hat{\xi}_M(n, l+1) = \hat{\xi}_{M-1}(n, l+1) - \frac{|v_{M-1}^{b(l+1)}(n)|^2}{B_{M-1}(n)} \tag{25}$$

2) Backward recursions for the error:

$$\hat{\xi}_{M-1}(n, l) = \hat{\xi}_M(n, l) + \frac{|v_{M-1}^{bl}(n)|^2}{B_{M-1}(n)} \tag{26}$$

$$\hat{\xi}_{M-1}(n-1, l) = \beta^{-1}\hat{\xi}_{M-1}(n, l) - \beta^{-1}\alpha_{M-1}^*(n, l) \cdot e_{M-1}(n, l) \tag{27}$$

$$\hat{\xi}_M(n, l-1) = \hat{\xi}_{M-1}(n-1, l) - \frac{|v_{M-1}^{f(l-1)}(n)|^2}{F_{M-1}(n)} \tag{28}$$

3) Forward recursions for the LS weight vector:

$$\hat{w}_{M-1}^{l+1}(n-1) = \lfloor \hat{w}_M^l(n) \rfloor_{M-1} + a_{M-1}(n)\hat{w}_{1M}^l(n) \tag{29}$$

$$\hat{w}_{M-1}^{l+1}(n) = \hat{w}_{M-1}^{l+1}(n-1) - \beta^{-1}g_{M-1}(n) \cdot e_{M-1}(n, l+1) \tag{30}$$

$$\hat{w}_M^{l+1}(n) = \begin{bmatrix} \hat{w}_{M-1}^{l+1}(n) \\ 0 \end{bmatrix} + \frac{v_{M-1}^{b(l+1)}(n)}{B_{M-1}(n)} \begin{bmatrix} -b_{M-1}(n) \\ 1 \end{bmatrix}. \tag{31}$$

[3]The notation $\lceil x \rceil_m$ stands for the vector made of the m top components of the vector x and $\lfloor x \rfloor_m$ for the vector made of the m bottom components of x.

4) Backward recursions for the LS weight vector:

$$\hat{w}_{M-1}^l(n) = \lceil \hat{w}_M^l(n) \rceil_{M-1} + b_{M-1}(n)\,\hat{w}_{MM}^l(n) \tag{32}$$

$$\hat{w}_M^{l-1}(n) = \begin{bmatrix} 0 \\ \hat{w}_{M-1}^l(n-1) \end{bmatrix} + \frac{v_{M-1}^{f(l-1)}(n)}{F_{M-1}(n)} \begin{bmatrix} 1 \\ -a_{M-1}(n) \end{bmatrix} \tag{33}$$

The forward recursions involve the following pattern: the quantity of interest is first expressed at time $n-1$ and for a lag of $l+1$, using a forward prediction of order $M-1$. Then the required quantity for a lag $l+1$ is computed by using a time update recursion and a backward prediction of order $M-1$.

III. The Joint RLS Algorithm

Based on the error and weight vector recursions, different variants of joint time delay and fast transversal filter algorithms can be obtained. These algorithms can be cast into the following general form:

1) Apply the RLS algorithm in order to obtain $\hat{w}_M^l(n)$ and $\hat{\xi}_M(n, l)$

2) Adapt l by using derivative information from $\hat{\xi}_M(n, l)$, and update $\hat{w}_M^l(n)$ and $\hat{\xi}_M(n, l)$ in the lag direction.

The joint tracking algorithm is composed of three distinct computational phases. The first part is essentially the preliminary phase of the FTF algorithm, i.e., the computations making use of the forward and backward linear predictors as well as the Kalman gain filter (see [8] and [5] for more details about the FTF or the FAEST algorithms). We use a slight modification of the conventional expression for the FTF algorithm by describing it in terms of $(M-1)$th-order predictors instead of the more conventional Mth-order. The second computational phase involves the computation of the current weight vector $\hat{w}_M^l(n)$ and the computation of three sums of errors; one for each element of the set of lags $\{l-1, l, l+1\}$. The computations of $\hat{\xi}_m(n, l)$, $\hat{\xi}_M(n, l+1)$ and $\hat{\xi}_M(n, l-1)$ are performed by using the lag update recursions for the error and the weight vector. In the joint algorithms considered in this paper, the computation of $\hat{w}_M^{l-1}(n)$ and $\hat{\xi}_M(n, l-1)$ is first performed, using the usual FTF equations. Then the forward lag recursions for both the error and the weight vector are used twice, in order to get the errors for l and $l+1$ and the weight vector for l. These successive applications of the forward recursions produce the least number of computations, compared for example to the application of the forward and backward recursions on the error and weight vector at lag l. This choice also simplifies the third computational phase, which involves a decision on the lag update and the computations of the new corresponding variables. The lag is either kept at the same value or updated by plus or minus one sample, depending on the largest of the three sums of errors $\{\hat{\xi}_M(n, l), \hat{\xi}_M(n, l+1), \hat{\xi}_M(n, l-1)\}$. Therefore, the

adaptation is not only done in the direction of the least squares solution, as in a gradient-type algorithm, but is such that a value that truly minimizes $\mathcal{E}(n)$ at each iteration, within a finite set of possible delay values, is selected. This type of joint algorithm computes the two estimates such that they correspond to the joint LS solution at each iteration.

Schematically, the preliminary and error computations phases of the algorithm can be represented as in Fig. 4, where six parallel digital filter are represented. The top three filters are essentially the same as the ones used in the conventional fast transversal filter [6], [4], except for the difference in predictors order. The fourth filter is for the computation of $\hat{\xi}_M(n, l-1)$ and $\hat{w}_M^{l-1}(n-1)$. Notice that $\hat{\xi}_{M-1}(n-1, l)$ is also obtained from that filter, using (23) and the fact that

$$\hat{w}_1^l(n) = \frac{v_{M-1}^{fl}(n)}{F_{M-1}(n)}. \tag{34}$$

A fifth filter, with weight vector $\hat{w}_{M-1}^l(n-1)$ obtained from (29), is used to obtain $v_{M-1}^{bl}(n)$, from which $\hat{\xi}_M(n, l)$, $\hat{w}_M^l(n)$ and $\hat{\xi}_{M-1}(n, l+1)$ are computed. Finally a sixth transversal filter, with weight vector $\hat{w}_{M-1}^{l+1}(n-1)$, is used in the computation of $v_{M-1}^{b(l+1)}(n)$ and $\hat{\xi}_M(n, l+1)$.

The joint algorithm, based on Fig. 4, is given in the next subsection. Parts a) and b) of this algorithm correspond to the figure, while part c) constitutes the lag update section. The decision about this update may involve the time average of the sum of squared errors or another form of average. In our algorithm, we choose to update the lag as

$$l = l + \hat{\nabla}_l \hat{\xi}_M(n, l). \tag{35}$$

where

$$\hat{\nabla}_l \hat{\xi}_M(n, l) = \begin{cases} 1, & \text{if } \langle \hat{\xi}_M(n, l+1) \rangle < \langle \hat{\xi}_M(n, l) \rangle \\ & \quad \text{and } \langle \hat{\xi}_M(n, l+1) \rangle < \langle \hat{\xi}_M(n, l-1) \rangle \\ -1, & \text{if } \langle \hat{\xi}_M(n, l-1) \rangle < \langle \hat{\xi}_M(n, l) \rangle \\ & \quad \text{and } \langle \hat{\xi}_M(n, l-1) \rangle < \langle \hat{\xi}_M(n, l+1) \rangle \\ 0, & \text{otherwise} \end{cases} \tag{36}$$

and $\langle \cdot \rangle$ denotes a form of time average.

Note that in the case of positive update, in (42), only a simple transfer of information from $l+1$ to l quantities and the reinitialization of certain variables, are required. In the case of negative update, in (43), some intermediate computations, involving $\theta_M^{l-1}(n)$ and $\theta_M^{l-2}(n)$, are necessary to obtain $v_{M-1}^{bl}(n)$ and $v_{M-1}^{f(l-2)}(n)$. The vectors $\theta_M^{l-1}(n)$ and $\theta_M^{l-2}(n)$ are updated in (38) and are referred to as "recursions for update smoothness." These quantities are used with some of the backward lag-recursive relations, in the computation of the new values of $w_M^{l-1}(n)$ and $\hat{\xi}_M(n, l-1)$.

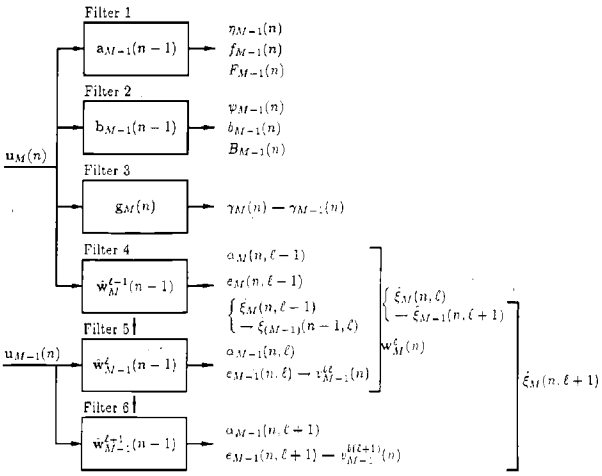


Fig. 4. Interpretation of the lag $l-1$, l and $l+1$ error computations, in terms of transversal filters.

A. The Joint Algorithm

A form of initial parameter acquisition is first discussed in this section, and the joint tracking algorithm is presented.

1) The Acquisition Procedure: The lag-recursive relations of Section II-B can be used in the acquisition algorithm, where it is required that the least squares solution be attained, with respect *to both the lag value and the optimum weight vector*. Since such an acquisition is to be performed on a finite set of data, it is in fact identical to the *optimum lag* problem considered in [7]. It is desired to compute, for a representative number of lag values, the least sum of squared errors and determine the lag and the weight vector corresponding to the global minimum. This can be done in various ways, with the help of the lag recursions. One method consists in first computing the least squares solution for the smallest lag value of interest. Then the computations of (40) can be used as many times as desired to obtain the sums of squared errors for all the lags of interest. The lag value corresponding to the least $\hat{\xi}_M(n, l)$ is retained for the initial value.

2) The Joint Tracking Algorithm: Based on the acquired lag and weight vector values, the following tracking algorithm is applied on the incoming data.

a) Preliminary computations:

$$\text{filter 1}\begin{cases}\eta_{M-1}(n) = u(n) - \boldsymbol{a}_{M-1}^H(n-1)\boldsymbol{u}_{M-1}(n-1)\\ f_{M-1}(n) = \dfrac{\eta_{M-1}(n)}{\gamma_{M-1}(n-1)}\\ \boldsymbol{a}_{M-1}(n) = \boldsymbol{a}_{M-1}(n-1) + \beta^{-1}\boldsymbol{g}_{M-1}(n-1)f_{M-1}^*(n)\\ F_{M-1}(n) = \beta F_{M-1}(n-1) + \eta_{M-1}(n)f_{M-1}^*(n)\end{cases}$$

$$\text{filter 3}\begin{cases}\boldsymbol{g}_M(n) = \begin{bmatrix}0\\ \boldsymbol{g}_{M-1}(n-1)\end{bmatrix} + \dfrac{\eta_{M-1}(n)}{F_{M-1}(n-1)}\begin{bmatrix}1\\ -\boldsymbol{a}_{M-1}(n-1)\end{bmatrix}\\ \boldsymbol{g}_{M-1}(n) = \lceil \boldsymbol{g}_M(n)\rceil_{M-1} + g_{MM}(n)\,\boldsymbol{b}_{M-1}(n-1)\\ \gamma_M(n) = \gamma_{M-1}(n-1) + \dfrac{|\eta_{M-1}(n)|^2}{\beta F_{M-1}(n-1)}\end{cases} \tag{37}$$

$$\text{filter 2}\begin{cases}\psi_{M-1}(n) = g_{MM}(n)\,B_{M-1}(n-1)\\ \gamma_{M-1}(n) = \gamma_M(n) - \beta^{-1}g_{MM}(n)\,\psi_{M-1}^*(n)\\ b_{M-1}(n) = \dfrac{\psi_{M-1}(n)}{\gamma_{M-1}(n)}\\ \boldsymbol{b}_{M-1}(n) = \boldsymbol{b}_{M-1}(n-1) + \beta^{-1}\boldsymbol{g}_{M-1}(n)\,b_{M-1}^*(n)\\ B_{M-1}(n) = \beta B_{M-1}(n-1) + \psi_{M-1}(n)\,b_{M-1}^*(n)\end{cases}$$

b) Errors and weight vector computations: Extra recursions for update smoothness

$$\boldsymbol{\theta}_M^{l-1}(n) = \beta\boldsymbol{\theta}_M^{l-1}(n-1) + \boldsymbol{u}_M(n)r^*(n+l-1)$$
$$\boldsymbol{\theta}_M^{l-2}(n) = \beta\boldsymbol{\theta}_M^{l-2}(n-1) + \boldsymbol{u}_M(n)r^*(n+l-2) \tag{38}$$

Lag $l-1$ computations (filter 4)

$$\alpha_M(n, l-1) = r(n+l-1) - \boldsymbol{w}_M^{(l-1)H}(n-1)\,\boldsymbol{u}_M(n)$$
$$e_M(n, l-1) = \frac{\alpha_M(n, l-1)}{\gamma_M(n)}$$
$$\hat{\boldsymbol{w}}_M^{l-1}(n) = \hat{\boldsymbol{w}}_M^{l-1}(n-1) + \beta^{-1}\boldsymbol{g}_M(n)\,e_M^*(n, l-1)$$
$$\hat{\xi}_M(n, l-1) = \beta\hat{\xi}_M(n-1, l-1) + \alpha_M^*(n, l-1)\cdot e_M(n, l-1) \tag{39}$$

Lag l computations (filter 5)

$$\hat{\boldsymbol{w}}_{M-1}^l(n-1) = \lfloor\hat{\boldsymbol{w}}_M^{l-1}(n)\rfloor_{M-1} + \boldsymbol{a}_{M-1}(n)\,w_{1M}^{l-1}(n)$$
$$\alpha_{M-1}(n, l) = r(n+l) - \hat{\boldsymbol{w}}_{M-1}^{lH}(n-1)\,\boldsymbol{u}_{M-1}(n)$$
$$e_{M-1}(n, l) = \frac{\alpha_{M-1}(n, l)}{\gamma_{M-1}(n)}$$
$$\hat{\boldsymbol{w}}_{M-1}^l(n) = \hat{\boldsymbol{w}}_{M-1}^l(n-1) + \beta^{-1}\boldsymbol{g}_{M-1}(n)\cdot e_{M-1}^*(n, l)$$
$$v_{M-1}^{bl}(n) = \beta v_{M-1}^{bl}(n-1) + \psi_{M-1}(n)\,e_{M-1}^*(n, l)$$
$$\hat{\boldsymbol{w}}_M^l(n) = \begin{bmatrix}\hat{\boldsymbol{w}}_{M-1}^l(n)\\ 0\end{bmatrix} + \frac{v_{M-1}^{bl}(n)}{B_{M-1}(n)}\begin{bmatrix}-\boldsymbol{b}_{M-1}(n)\\ 1\end{bmatrix}$$
$$\hat{\xi}_{M-1}(n-1, l) = \hat{\xi}_M(n, l-1) + F_{M-1}(n)|w_{1M}^{l-1}(n)|^2$$
$$\hat{\xi}_{M-1}(n, l) = \beta\hat{\xi}_{M-1}(n-1, l) + \gamma_{M-1}(n)\cdot|e_{M-1}(n, l)|^2$$
$$\hat{\xi}_M(n, l) = \hat{\xi}_{M-1}(n, l) - \frac{|v_{M-1}^{bl}(n)|^2}{B_{M-1}(n)} \tag{40}$$

Lag $l+1$ computations (filter 6)

$$\hat{\boldsymbol{w}}_{M-1}^{l+1}(n-1) = \lfloor\hat{\boldsymbol{w}}_M^l(n)\rfloor_{M-1} + \boldsymbol{a}_{M-1}(n)\cdot w_{1M}^l(n)$$
$$\alpha_{M-1}(n, l+1) = r(n+l+1) - \hat{\boldsymbol{w}}_{M-1}^{(l+1)H}(n-1)\cdot\boldsymbol{u}_{M-1}(n)$$
$$e_{M-1}(n, l+1) = \frac{\alpha_{M-1}(n, l+1)}{\gamma_{M-1}(n)}$$
$$\hat{\boldsymbol{w}}_{M-1}^{l+1}(n) = \hat{\boldsymbol{w}}_{M-1}^{l+1}(n-1) + \beta^{-1}\boldsymbol{g}_{M-1}(n)\cdot e_{M-1}^*(n, l+1)$$
$$v_{M-1}^{b(l+1)}(n) = \beta v_{M-1}^{b(l+1)}(n-1) + \psi_{M-1}(n)\cdot e_{M-1}^*(n, l+1)$$
$$\hat{\xi}_{M-1}(n-1, l+1) = \hat{\xi}_M(n, l) + F_{M-1}(n)|w_{1M}^l(n)|^2$$

$$\hat{\xi}_{M-1}(n, l+1) = \beta\hat{\xi}_{M-1}(n-1, l+1) + \gamma_{M-1}(n) \cdot |e_{M-1}(n, l+1)|^2$$

$$\hat{\xi}_M(n, l+1) = \hat{\xi}_{M-1}(n, l+1) - \frac{|v_{M-1}^{b(l+1)}(n)|^2}{B_{M-1}(n)}. \quad (41)$$

c) Updates: **If** $\langle\hat{\xi}_M(n, l+1)\rangle < \langle\hat{\xi}_M(n, l)\rangle$ **and** $\langle\hat{\xi}_M(n, l+1)\rangle < \langle\hat{\xi}_M(n, l-1)\rangle$ **then**

$$\begin{aligned}
\hat{w}_M^{l-1}(n) &\leftarrow \hat{w}_M^{l}(n) \\
\hat{\xi}_M(n, l-1) &\leftarrow \hat{\xi}_M(n, l) \\
v_{M-1}^{bl}(n) &\leftarrow v_{M-1}^{b(l+1)}(n) \\
\theta_M^{l-2}(n) &\leftarrow \theta_M^{l-1}(n) \\
\theta_M^{l-1}(n) &= u_M(n)r^*(n+l) \\
v_{M-1}^{b(l+1)}(n) &= 0 \\
l &\leftarrow l+1
\end{aligned} \quad (42)$$

Endif

If $\langle\hat{\xi}_M(n, l-1)\rangle < \langle\hat{\xi}_M(n, l)\rangle$ **and** $\langle\hat{\xi}_M(n, l-1)\rangle < \langle\hat{\xi}_M(n, l+1)\rangle$ **then**

$$v_{M-1}^{b(l-1)}(n) \leftarrow v_{M-1}^{bl}(n)$$

$$v_{M-1}^{bl}(n) = [-b_{M-1}^H(n) \quad 1]\,\theta_M^{l-1}(n)$$

$$v^{f(l-2)}_{M-1}(n) = [1 \; - a_{M-1}^H(n)]\,\theta_M^{l-2}(n)$$

$$\hat{\xi}_{M-1}(n, l-1) = \hat{\xi}_M(n, l-1) + B_{M-1}(n)|w_{MM}^{l-1}(n)|^2$$

$$\hat{w}_{M-1}^{l-1}(n-1) = \lceil\hat{w}_M^{l-1}(n-1)\rceil_{M-1} + b_{M-1}(n-1) \cdot w_{MM}^{l-1}(n-1)$$

$$\hat{w}_M^{l-1}(n) = \begin{bmatrix} 0 \\ \hat{w}_{M-1}^{l-1}(n-1) \end{bmatrix} + \frac{v^{f(l-2)}_{M-1}(n)}{F_{M-1}(n)} \cdot \begin{bmatrix} 1 \\ -a_{M-1}(n) \end{bmatrix}$$

$$\hat{\xi}_{M-1}(n-1, l-1) = \beta^{-1}\hat{\xi}_{M-1}(n, l-1) - \beta^{-1}\frac{|u_{M-1}^H(n)\,\hat{w}_{M-1}^{l-1}(n-1) - r^*(n+l-1)|^2}{\gamma_{M-1}(n)}$$

$$\hat{\xi}_M(n, l-1) = \hat{\xi}_{M-1}(n-1, l-1) - \frac{|v^{f(l-2)}_{M-1}(n)|^2}{F_{M-1}(n)}$$

$$\begin{aligned}
\theta_M^{l-1}(n) &\leftarrow \theta_M^{l-2}(n) \\
\theta_M^{l-2}(n) &= u_M(n)r^*(n+l-3) \\
l &\leftarrow l-1
\end{aligned} \quad (43)$$

Endif ■

B. Discussion

The originality of the joint tracking LS algorithm presented in Section III-A2) resides in the serial computations, from $\hat{w}_M^{l-1}(n-1)$, of all the necessary errors and weight vectors for lags l and $l+1$. One consequence of this serial approach is a reduction in the memory needed to store the different quantities of interest. The lag-update recursions append themselves nicely to the FTF algorithm. Note however that two extra recursions (38) are necessary to ensure smoothness when the lag is updated from l to $l-1$ (43). In this case, the cross-correlation vectors $\theta_M^{l-1}(n)$ and $\theta_M^{l-2}(n)$ are necessary to update $v_{M-1}^{bl}(n)$ and to compute $v^{f(l-2)}_{M-1}(n)$ (necessary to correct $\hat{w}_M^{l-1}(n)$). Note also that $\theta_M^{l-1}(n)$, $\theta_M^{l-2}(n)$, and $v_{M-1}^{b(l+1)}(n)$ must be reinitialized in the case of lag update [(42) and (43)]. These reinitializations constitute the only approximations of the joint LS algorithm and are justified by the limited memory of the algorithm (defined by β). Furthermore, the reinitialization of the cross-correlation vectors does not involve any of the algorithm's internal variables since the input signal $u(n)$ and the reference signal $r(n)$ are the only variables used in these computations.

In contrast, the application of three parallel versions of the RLS algorithm, one for each possible lag, requires the initialization of both the sum of squared errors and the weight vector, when the lag is corrected. The initialization must be done assuming *a priori* knowledge about the previous input data (usually zero data are assumed). This typically introduces an error in both $\hat{\xi}_M^l(n)$ and $\hat{w}_M^l(n)$ because their computation involves the internal variables $\gamma_M(n)$ and $g_M(n)$ [see (39)] that were obtained from a totally different set of initial conditions (nonzero input data). In order to allow a smooth transition in the case of lag update, two extra parallel branches, one for $l+2$ and one for $l-2$, must be computed, which gives a final parallel algorithm involving five branches. This algorithm requires approximately five times the amount of memory needed by the algorithm based on the lag-recursive relations (in order to store all the previous values of the variables used in the errors and weight vectors computation).

At the start of the joint algorithm, the internal variables of the FTF are initialized *exactly* as proposed by Cioffi [6], and the extra error and correlation variables are initialized to zero.

IV. Experimental Results

The fast joint time delay estimation and adaptive RLS filtering algorithm corresponding to Fig. 4 was implemented, in order to verify its practical behavior. The reference filter $h(n)$ was implemented as a 21-tap low-pass FIR filter, with a 3-dB bandwidth approximately equal to 0.7π. The transmitted signal $s(n)$ was a zero-mean process with a white power spectral density from $-\pi$ to π. The adaptive filter also had 21 taps. The usual problem of numerical instability, often associated with the RLS algorithm implementations [6], was also present in our

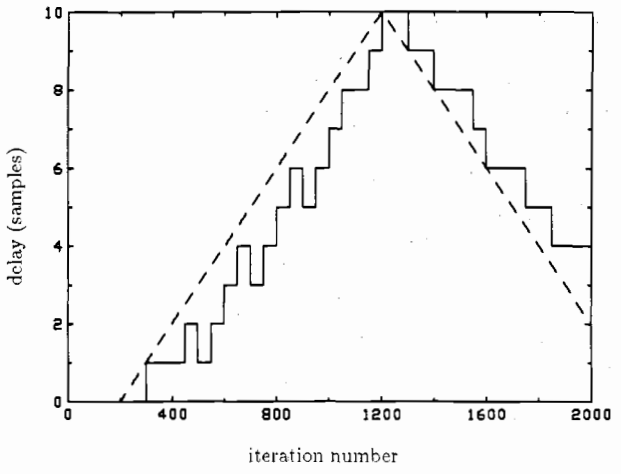


Fig. 5. Tracking of a linearly changing delay; ------ reference delay, —— adaptive delay. $\beta = 0.92$, noiseless conditions.

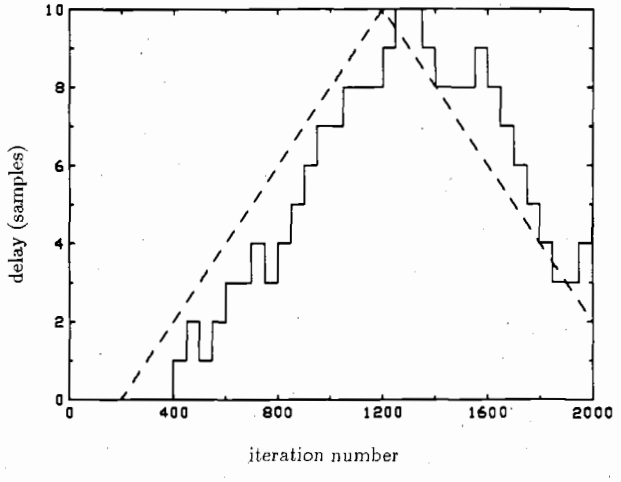


Fig. 6. Tracking of a linearly changing delay; ------ reference delay, —— adaptive delay. $\beta = 0.92$, SNR = 20 dB.

algorithm. A periodic restart of the algorithm, similar to the technique proposed by Eleftheriou and Falconer [9], was implemented to reinitialize the internal variables of the algorithm (those produced by the first computational phase). Since we were only interested in the joint delay estimation and adaptive filtering capabilities of the algorithm, our periodic restart procedure consisted in the simple periodic transfer of the parameters of an FTF algorithm, computed in parallel with the joint algorithm. The restart period was fixed to 600 iterations, a number large enough to ensure convergence in our particular case. The time when divergence occurs depends on the processor word length, on the nature of the specific algorithm and on the nature of the signal [10]. A real-time implementation should therefore make use of a *rescue variable* to monitor the truncation error accumulation in the algorithm and trigger the reinitialization of the critical variables [4].

Some simulation results concerning the tracking ability of the joint RLS algorithm are given in Figs. 5–8. In order to perform the lag-update decision [Part c) of Section III-A2)] the time average of the sum of squared errors must be computed. This is done by accumulating the sum of squared errors over 50 iterations.

The adaptive delay responses to a linearly changing reference delay are presented in Figs. 5 and 6. The reference slope is 0.01 sample/sampling period. The noiseless case is shown in Fig. 5, and the results for a signal-to-noise ratio of 20 dB appear in Fig. 6. Except for a granular-type of noise, the adaptive delay tracks well the reference delay. Note that the exponential window parameter β was set to 0.92, in order to allow good tracking. The results for a sinusoidal reference delay are illustrated in Figs. 7 and 8. Adequate tracking is again demonstrated in this case.

These results demonstrate the fact that the joint algorithm follows the reference delay closely, which indicates that the adaptive filter impulse response stays centered,

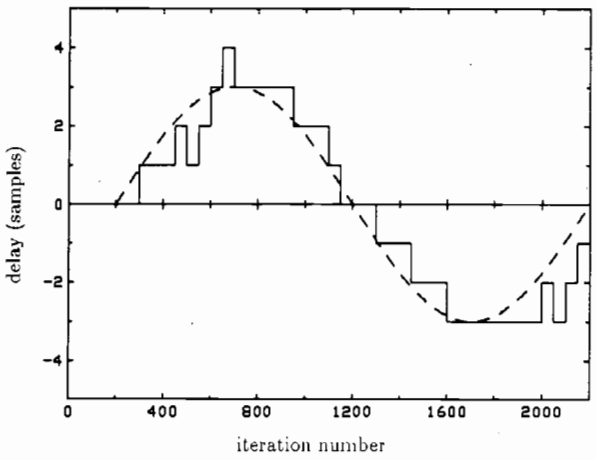


Fig. 7. Tracking of a sinusoidally changing delay; ------ reference delay, —— adaptive delay. $\beta = 0.92$, noiseless conditions.

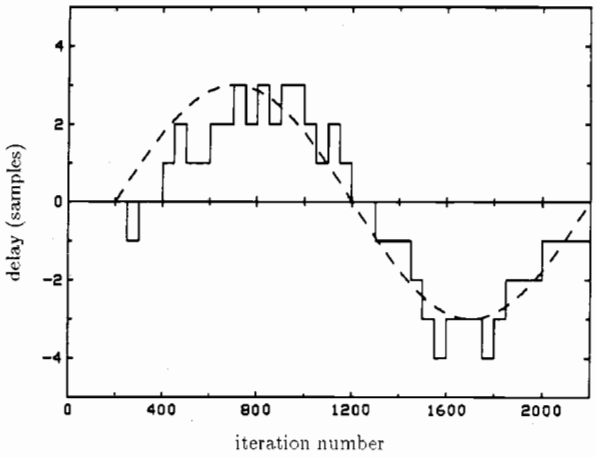


Fig. 8. Tracking of a sinusoidally changing delay; ------ reference delay, —— adaptive delay. $\beta = 0.92$, SNR = 20 dB.

even if the reference delay attains a fairly large value, as in the case of the linearly changing delay. This shows the potential of the joint algorithm in tempering significantly the problem of unwanted delay tracking by the adaptive filter.

V. Conclusion

This paper has presented a new algorithm for the joint estimation of time delays and correlation function between two observed signals, when the estimation criterion is the minimization of the sum of squared errors. The principal contribution of the work is the derivation of an RLS algorithm that makes use of time update, filter order and lag update relations, in order to compute efficiently the least squared error and the corresponding weight vector. The lag update constitutes an additional degree of freedom for the minimization of the sum of weighted squared errors.

The simulations of the joint RLS algorithm indicate the potential of the joint algorithm. By averaging the minimum sums of squared errors over 50 samples, and by comparing three of these sums of errors, the delay tracking is very good in all cases for SNR's as low as 20 dB. Below this value, the performances degrade very quickly. But for each application, there is an optimum strategy for delay estimation, and the particular one chosen here is fairly empirical. This simple method shows that the joint RLS algorithm can keep the adaptive filter impulse response approximately centered in different kinds of scenarios. It indicates also that if rapid adaptation to the reference filter is required and that computational complexity is a secondary issue, the conventional RLS adaptive filter can be favorably enhanced by the delay estimation based on the lag-recursive relations.

Acknowledgment

The computing equipment of Bell Northern Research, l'Institut National de la Recherche Scientifique, and the Communications Research Centre are greatly acknowledged. Comments from the various reviewers are appreciated.

References

[1] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 320–327, Aug. 1976.

[2] B. Champagne, M. Eizenman, and S. Pasupathy, "Exact maximum likelihood time delay estimation for short observation intervals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 39, no. 6, pp. 1245–1257, June 1991.

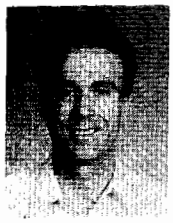
[3] J. A. Stuller, "Maximum-likelihood estimation of time-varying delay–part I," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 300–313, Mar. 1987.

[4] S. Haykin, *Adaptive Filter Theory*, Englewood Cliffs, NJ: Prentice-Hall, 1986.

[5] G. Carayannis, D. G. Manolakis, and N. Kalouptsidis, "A fast sequential algorithm for least-squares filtering and prediction," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 1394–1402, Dec. 1983.

[6] J. M. Cioffi and T. Kailath, "Fast, recursive-least-squares transversal filters for adaptive filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 304–337, Apr. 1984.

[7] N. Kalouptsidis, G. Carayannis, and D. G. Manolakis, "Fast design of multichannel FIR least-squares filters with optimum lag," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 48–59, Feb. 1984.
[8] D. Boudreau, "Joint time delay estimation and adaptive filtering techniques," Ph.D. dissertation, McGill University, 1990.
[9] E. Eleftheriou and D. D. Falconer, "Tracking properties and steady-state performance of RLS adaptive filter algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 1097–1110, Oct. 1986.
[10] P. Fabre and C. Gueguen, "Improvement of the fast recursive least-squares algorithms via normalization: A comparative study," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 296–308, Apr. 1986.



Daniel Boudreau received the B.Eng. degree from the University of Sherbrooke, Sherbrooke, Québec, Canada, the M.Eng. degree from Carleton University, Ottawa, Ontario, Canada, and the Ph.D. degree from McGill University, Montreal, Québec, Canada, in 1982, 1987, and 1990, respectively, all in electrical engineering.

From 1983 to 1987, he was with the Communications Processing Group of the Communications Research Centre (CRC), Ottawa, Canada, where he was involved in the design of the first generation of modems for the Canadian Mobile Satellite program. From 1987 to 1990, he was on educational leave from CRC, working toward the Ph.D. degree in the fields of delay estimation and adaptive filtering. Since 1990, he has been with the Mobile Satellite Group of the CRC, where he conducts research and development in the area of narrow-band modulation techniques for communications over fading channels.

Peter Kabal (S'70–M'75) received the B.A.Sc., M.A.Sc., and Ph.D. degrees in electrical engineering from the University of Toronto, Toronto, Ontario, Canada.

He is an Associate Professor in the Department of Electrical Engineering at McGill University, Montreal, Québec, Canada, and a Visiting Professor at INRS-Telecommunications (a research institute affiliated with the Université du Québec), Verdun, Québec. From September 1982 to September 1983 he was a full-time consultant to the Speech Communications Group at Bell-Northern Research, Verdun. From September 1987 to June 1988 he spent a sabbatical year as a Visiting Professor at the University of California, Santa Barbara. His current research interests focus on signal processing as applied to speech coding, adaptive filtering, and data transmission.