Signal Processing 18 (1989) 33-42
Elsevier Science Publishers B.V.

CORRECTING THE INSTABILITY DUE TO FINITE PRECISION OF THE FAST KALMAN IDENTIFICATION ALGORITHMS

George V. MOUSTAKIDES
Computer Technology Institute, P.O. Box 1122, 26110 Patras, Greece

Received 28 March 1988
Revised 26 January 1989

Abstract. The effect of accumulation of round-off errors, leading eventually to instability, limits the use of exponentially weighted fast Kalman least-squares identification algorithms. With the present paper we try to eliminate this severe drawback of this class of algorithms. The idea for deriving a more stable version is similar to an existing work, our derivation though being more elegant and leading to an algorithm with significantly lower complexity. Specifically, the proposed algorithm has complexity $8N$, where N is the number of parameters to be estimated. This is only N more multiplications, compared to the fastest Kalman Algorithm. The stabilization is achieved by correcting in a very well defined manner the two prior (forward and backward) residuals.

Zusammenfassung. Die Akkumulation von Rundungsfehlern, die in ungünstigen Fällen sogar zur Instabilität führen kann, begrenzt die Leistungsfähigkeit exponentiell gewichteter schneller Kalman-Algorithmen nach dem Prinzip des kleinsten Fehlerquadrats. In vorliegendem Beitrag versuchen wir, diese ungünstige Eigenschaft zu vermeiden. Die hier vorgeschlagene Methode der Stabilisierung lehnt sich an ein bereits bestehendes Verfahren an; unsere Ableitung ist jedoch eleganter und führt zu einem weniger aufwendigen Algorithmus. Die Komplexität des Algorithmus ergibt sich zu $8N$, wobei N die Zahl der zu schätzenden Parameter ist. Verglichen mit dem schnellsten Kalman-Algorithmus bedeutet dies nur N Multiplikationen mehr. Das Verfahren wird stabilisiert, indem die beiden Residualsignale (Vor- und Rückwärtsprädiktion) in wohldefinierter Weise korrigiert werden.

Résumé. Les effets de l'accumulation des erreurs d'arrondi, allant jusqu'à l'instabilité, limitent l'utilisation des algorithmes rapides de Kalman aux moindres carrés, à oubli exponentiel. Dans l'article présent, nous essayons d'éliminer cet important inconvénient de cette classe d'algorithmes. L'idée d'aboutir à une version plus stable est semblable à celle d'un travail existant; notre derivation étant plus élégante et conduisant à un algorithme de moindre complexité. L'algorithme proposé a une complexité $8N$, où N est le nombre de paramètres à estimer. Cela fait seulement N multiplications de plus que le plus rapide algorithme de Kalman. La stabilisation est assurée en corrigeant les deux résidus à priori (direct et rétrograde).

Keywords. Recursive estimation, fast least squares.

1. Introduction

Limited precision effects, severely degrade the performance of certain fast recursive least-squares identification algorithms. Actually the faster the version the larger the degradation. The accumulation of round-off errors leads either to a blow to infinity of the variables of the algorithm or limits the tracking capability of the algorithm by making its gain very small [2]. In [4, 18] we can find overviews of the most popular versions of least-squares and in [7] their performance with respect to limited precision errors.

One type of least-squares algorithms that suffer from instability caused by round-off errors are the exponentially weighted fast Kalman algorithms (FKA). There are several versions of FKA [5, 6, 8, 9, 13, 14] with different complexities and

all of them suffer from instability. A useful classification of FKA can be found in [17]. The fastest versions [5, 8] have complexity $7N$, with N the order of the filter to be estimated. These versions are highly unstable.

Even though the instability of the FKA is an obvious fact in practice, proving it theoretically is very hard. In [16] it was shown that the FKA version of [14, 15] is unstable for a first order filter and for a specific input. In [2], with certain assumptions on the input signal and the rounding errors of the filters, the instability of the FKA versions of [5, 8] was shown. Several techniques were used in order to stabilize the FKA [1, 3, 11, 12]. Most of them yield algorithms that are not realizations of the least-squares, thus having a reduced tracking capability.

In this paper we will concentrate on the technique introduced in [3] and try to improve it. The idea in [3] was basically to compute the backward prior residual in two independent ways. The difference of these two ways was regarded as a measure for the accumulation of the round-off errors. Consequently this difference was used to correct the two already computed forward and backward filters. The resulting algorithm had complexity $10N$ and for its derivation certain approximations were necessary. We will use the same idea as in [3]. We will compute the backward prior residual in two different ways and use the difference to stabilize the algorithm. Instead of correcting through the computed filters we will try to directly affect the computation of the filters of the next time instant. This will lead to a derivation of an algorithm without any approximations and of complexity $8N$. This is only N more multiplications compared to the fastest FKA (versions FAEST, FTF; actually there are versions of FAEST and FTF that have complexity $8N$), but the resulting algorithm has a much more stable behavior. The additional N multiplications are used to compute the backward residual in the normal way, by multiplying the backward filter with the input vector.

2. Derivation of the algorithm

We will present our method by directly applying it to the FAEST version of the FKA of [5]. Before deriving the algorithm we will introduce some notations and some basic relations from the least-squares theory that we are going to use. With lower case letters we denote scalars and with upper case vectors. Vectors will have two indices; the first denoting their length and the second the time. If X is a vector then X^i denotes the ith element of the vector X. Barred variables will denote corrected versions of the same variables. Finally "′" denotes transpose.

The recursive least-squares identification algorithms solve the following problem: Given sequentially two sequences $\{x(T)\}$ and $\{y(T)\}$ estimate at every time instant T a filter $H_{N,T}$ that minimizes $v_N(T)$ defined by

$$v_N(T)=\sum_{k=0}^{T}\lambda^{T-k}[y(k)-H'_{N,T}X_{N,T}]^2, \qquad (1)$$

where $0<\lambda\leq 1$ is the exponential forgetting factor, $X_{N,T}=[x(T),\ldots,x(T-N+1)]'$ with $x(T)=y(T)=0$ for $T\leq 0$. Given the solution $H_{N,T-1}$ at time $T-1$ of the minimization, the solution at time T can be computed from the recursion (see [15])

$$\begin{aligned} e_N(T)&=y(T)-H'_{N,T-1}X_{N,T},\\ \varepsilon_N(T)&=\gamma_N(T)e_N(T),\\ H_{N,T}&=H_{N,T-1}-\varepsilon_N(T)W_{N,T}, \end{aligned} \qquad (2)$$

where $e_N(T)$, $\varepsilon_N(T)$ are the prior and posterior estimation residuals and $W_{N,T}$ is known as the Dual Kalman Gain. It was the use of the Dual Kalman Gain that resulted in the derivation of the fastest FKA (introduced in [5] and resulted in the version FAEST with complexity $7N$). The quantities $W_{N,T}$ and $\gamma_N(T)$ are defined by the following equations

$$W_{N,T}=-\frac{1}{\lambda}R^{-1}_{N,T-1}X_{N,T}, \qquad (3)$$

with

$$R_{N,T} = \sum_{k=0}^{T} \lambda^{T-k} X_{N,k} X'_{N,k} \tag{4}$$

and

$$\gamma_N(T) = \frac{1}{1 - W'_{N,T} X_{N,T}}. \tag{5}$$

It is easy to see that $0 \leq \gamma_N(T) \leq 1$. All FKA efficiently update $W_{N,T}$ and $\gamma_N(T)$ requiring a number of operations proportional to N, with the fastest versions requiring $5N$ multiplications (FAEST, FTF [5, 8]). Combining this with the relations in (2), we can update $H_{N,T}$ with a total of $7N$ multiplications per time step. This is the fastest version of FKA (and the fastest realization of the least-squares for a given order). As we said, we will present our method by applying it directly to FAEST. Table 1 contains in detail all the steps of the FAEST algorithm. Notice that the algorithm of Table 1 differs slightly from the conventional FAEST of [5], in the sense that we prefer to propagate $\gamma_N(T)$ of (5) instead of the inverse of this variable propagated in [5]. The reason is that with this variable we will avoid some divisions by replacing them with multiplications. We will give

Table 1
The fast a-posteriori error sequential technique (FAEST)—complexity $7N$

	Multiplications	Divisions
Available at time T: $W_{N,T-1}, A_{N,T-1}, B_{N,T-1}, H_{N,T-1}, X_{N,T-1}, \gamma_N(T-1), \alpha_N(T-1), \beta_N(T-1)$.		
New information: $y(T), x(T)$.		
Time update of the Kalman Gain $W_{N,T}$		
$e^{f}_N(T) = x(T) - A'_{N,T-1} X_{N,T-1}$	N	—
$\varepsilon^{f}_N(T) = \gamma_N(T-1) e^{f}_N(T)$	1	—
$\alpha_N(T) = \lambda \alpha_N(T-1) + e^{f}_N(T) \varepsilon^{f}_N(T)$	2	—
$\gamma_{N+1}(T) = \frac{\lambda \alpha_N(T-1)}{\alpha_N(T)} \gamma_N(T-1)$	2	1
$W_{N+1,T} = \begin{bmatrix} 0 \\ W_{N,T-1} \end{bmatrix} - \frac{e^{f}_N(T)}{\lambda \alpha_N(T-1)} \begin{bmatrix} 1 \\ -A_{N,T-1} \end{bmatrix}$	$N+1$	1
$A_{N,T} = A_{N,T-1} - \varepsilon^{f}_N(T) W_{N,T-1}$	N	—
$e^{b}_N(T) = -\lambda \beta_N(T-1) W^{N+1}_{N+1,T}$	2	—
$\theta_N(T) = 1 + \gamma_{N+1}(T) e^{b}_N(T) W^{N+1}_{N+1,T}$	2	—
$\gamma_N(T) = \frac{\gamma_{N+1}(T)}{\theta_N(T)}$	—	1
$\varepsilon^{b}_N(T) = \gamma_N(T) e^{b}_N(T)$	1	—
$\beta_N(T) = \lambda \beta_N(T-1) + e^{b}_N(T) \varepsilon^{b}_N(T)$	2	—
$\begin{bmatrix} W_{N,T} \\ 0 \end{bmatrix} = W_{N+1,T} - W^{N+1}_{N+1,T} \begin{bmatrix} -B_{N,T-1} \\ 1 \end{bmatrix}$	N	—
$B_{N,T} = B_{N,T-1} - \varepsilon^{b}_N(T) W_{N,T}$	N	—
Time update of the filter $H_{N,T}$		
$e_N(T) = y(T) - H'_{N,T} X_{N,T}$	N	—
$\varepsilon_N(T) = \gamma_N(T) e_N(T)$	1	—
$H_{N,T} = H_{N,T-1} - \varepsilon_N(T) W_{N,T}$	N	—
Total number of multiplications/divisions	$7N+14$	3

one more relation that we will use and that can be easily shown to be true using the matrix inversion lemma,

$$R_{N,T}^{-1}X_{N,T} = -\gamma_N(T)\,W_{N,T}. \tag{6}$$

We will now proceed to the derivation of the algorithm. As in [3] we concentrate only in stabilizing the part concerning the forward and backward filters $A_{N,T}$, $B_{N,T}$. Notice from Table 1, that the prior backward residual $e_N^b(T)$ is computed using the following equation

$$e_N^b(T) = -\lambda\beta_N(T-1)\,W_{N,T-1}^N - \frac{\beta_N(T-1)}{\alpha_N(T-1)}\,A_{N,T-1}^N\,e_N^f(T), \tag{7}$$

where $e_N^f(T)$ is the prior forward residual. There is another way of computing $e_N^b(T)$, that is, in the normal way

$$e_N^b(T) = x(T-N) - B'_{N,T-1}X_{N,T}. \tag{8}$$

Following now the idea of [3] we define the difference of (7) and (8) as

$$\xi_N(T) = e_N^b(T) + k_N(T-1)e_N^f(T) + \lambda\beta_N(T-1)\,W_{N,T-1}^N, \tag{9}$$

where $e_N^b(T)$ denotes the computation in (8) and $k_N(T-1) = \beta_N(T-1)A_{N,T-1}^N/\alpha_N(T-1)$. The quantity in (9) under infinite precision is always equal to zero and under limited precision can be regarded as a measure of accumulation of round-off errors. That $\xi_N(T)$ is indeed an efficient measure can be seen from the simulations. In Fig. 1, for example, it is plotted for the case $N=20$. Its magnitude gradually increases with time, indicating the accumulation of the round-off errors, while $\gamma_N(T)$ has acceptable values (in the interval [0, 1]). At some point $\gamma_N(T)$ exceeds unity or becomes negative and the algorithm is reinitialized (this reinitialization procedure is used in [8]). Notice that $\xi_N(T)$ is a function of $A_{N,T-1}$ and $B_{N,T-1}$; this is the reason why in [3] the algorithm corrects these filters. We would like to express

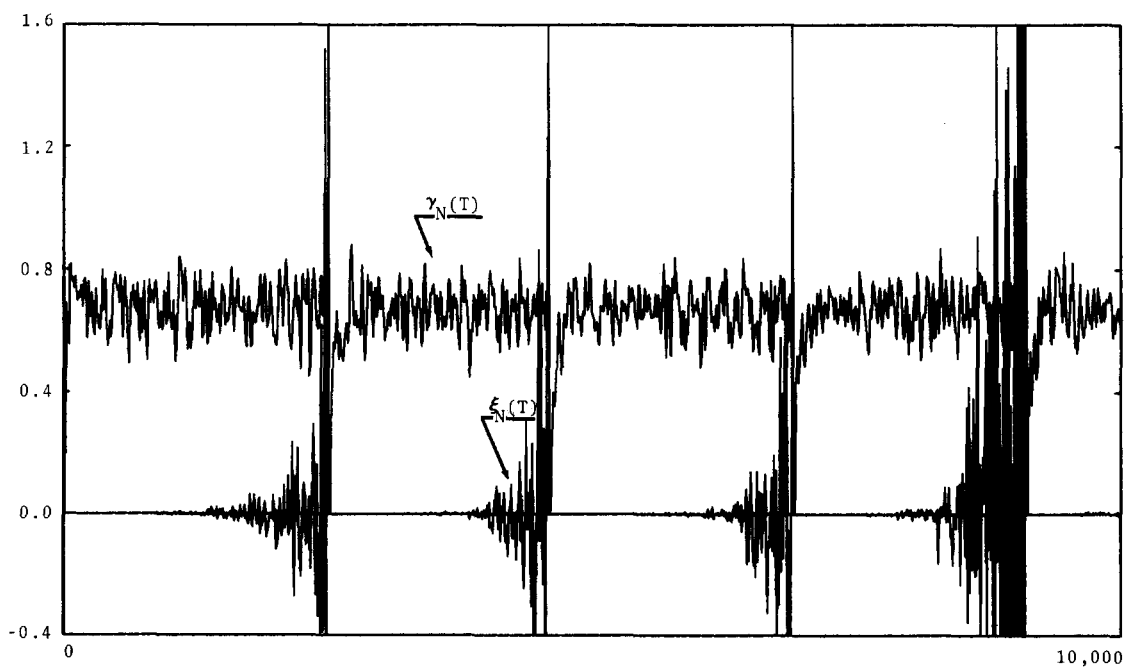

Fig. 1. Performance of the FAEST algorithm for $N=20$ and for white noise data sequence.

$\xi_N(T)$ in terms of $A_{N,T}$ and $B_{N,T}$ in order to correct directly the filters of the next time instant. We can easily introduce these filters by writing the prior residuals in terms of the posterior, specifically

$$\xi_N(T)=\frac{\varepsilon_N^b(T)}{\gamma_N(T)}+k_N(T-1)\frac{\varepsilon_N^f(T)}{\gamma_N(T-1)}+\lambda\beta_N(T-1)W_{N,T-1}^N \tag{10}$$

and express in (10) the two posterior residuals in terms of the filters $A_{N,T}$, $B_{N,T}$ as follows

$$\xi_N(T)=\frac{1}{\gamma_N(T)}[x(T-N)-B'_{N,T}X_{N,T}]+\frac{k_N(T-1)}{\gamma_N(T-1)}[x(T)-A'_{N,T}X_{N,T}]+\lambda\beta_N(T-1)W_{N,T-1}^N. \tag{11}$$

The corrections will be made in such a way that if in (11) we substitute the filters with their corrections $\bar{A}_{N,T}$ and $\bar{B}_{N,T}$ the resulting $\xi_N(T)$ is minimized. Let us call $\bar{\xi}_N(T)$ the quantity in (12) but with the two filters replaced with their corrections. We can then write

$$\bar{\xi}_N(T)=\xi_N(T)-\frac{1}{\gamma_N(T)}\Delta B'X_{N,T}-\frac{k_N(T-1)}{\gamma_N(T-1)}\Delta A'X_{N,T-1}, \tag{12}$$

where $\Delta A=\bar{A}_{N,T}-A_{N,T}$, $\Delta B=\bar{B}_{N,T}-B_{N,T}$ and $\xi_N(T)$ denotes the quantity in (9). We would like to minimize $\bar{\xi}_N(T)$ but at the same time keep small the values of the corrections ΔA and ΔB. These requirements are expressed by the minimization of the following criterion

$$w_N(T)=\frac{1}{\gamma_N(T-1)}\Delta A'R_{N,T-1}\Delta A+\frac{1}{\gamma_N(T)}\Delta B'R_{N,T}\Delta B+\rho[\bar{\xi}_N(T)]^2, \tag{13}$$

where ρ is a constant. As we will see from the derivation, in order to correct the two filters it will be enough to correct the forward and backward residuals. The reason for dividing the first two terms with the corresponding γ_N is only to make the corrections to the prior residuals instead of the posterior. We could, without any problem, replace them with unities. Notice that

$$\Delta A'R_{N,T-1}\Delta A=\sum_{k=0}^{T}\lambda^{T-k}[(x(k)-\bar{A}'_{N,T}X_{N,k-1})-(x(k)-A'_{N,T}X_{N,k-1})]^2,$$

which is the norm of the difference of the two (length T) error vectors. In other words we would like the corrections to be small in the sense that the two resulting error vectors are close to each other. This definition of distance is preferable from the Euclidian distance of the filter and its correction, because it leads to corrections that are scale invariant; a property that also has the two filters $A_{N,T}$ and $B_{N,T}$. Minimizing (13) is straight-forward and needs no approximations as was the case in [3]. The solution, using (6), is given by

$$\Delta A=\rho\bar{\xi}_N(T)k_N(T-1)R_{N,T-1}^{-1}X_{N,T-1}=-\gamma_N(T-1)\rho\bar{\xi}_N(T)k_N(T-1)W_{N,T-1}, \tag{14}$$

$$\Delta B=\rho\bar{\xi}_N(T)R_{N,T}^{-1}X_{N,T}=-\gamma_N(T)\rho\bar{\xi}_N(T)W_{N,T}, \tag{15}$$

where

$$\bar{\xi}_N(T)=\left\{1+\rho\left(\frac{1}{\gamma_N(T)}-1\right)+\rho(k_N(T-1))^2\times\left(\frac{1}{\gamma_N(T-1)}-1\right)\right\}^{-1}\xi_N(T). \tag{16}$$

Using (14) and (15) and Table 1 we can express $\bar{A}_{N,T}$, $\bar{B}_{N,T}$ directly in terms of $A_{N,T-1}$ and $B_{N,T-1}$ as follows

$$\bar{A}_{N,T}=A_{N,T-1}-\gamma_N(T-1)\times(e_N^f(T)+k_N(T-1)\rho\bar{\xi}_N(T))W_{N,T-1}, \tag{17}$$

$$\bar{B}_{N,T} = B_{N,T-1} - \gamma_N(T) \times (e^{b}_N(T) + \rho\bar{\xi}_N(T))\, W_{N,T}. \quad (18)$$

Equations (17) and (18) show that we just have to introduce a correction in the two prior residuals in order to correct the filters of the time instant T. This is intuitively correct since the place where the rounding errors mainly accumulate is at the computation of the two prior residuals. To the corrected filters correspond corrected residuals. From (17) and (18), by multiplying correspondingly with $X_{N,T-1}$ and $X_{N,T}$ we can compute the corrections for the posterior residuals. Dividing these two residuals with the corresponding γ_N yields the corrected prior residuals. The result is given with the following equations

$$\bar{e}^{f}_N(T) = e^{f}_N(T) - \left(\frac{1}{\gamma_N(T-1)} - 1\right) \times k_N(T-1)\rho\bar{\xi}_N(T), \quad (19)$$

$$\bar{e}^{b}_N(T) = e^{b}_N(T) - \left(\frac{1}{\gamma_N(T)} - 1\right)\rho\bar{\xi}_N(T). \quad (20)$$

We can simplify our algorithm by avoiding some divisions introduced in (16, 19, 20). Since $\gamma_N(T)$ is usually close to unity we can use the following approximation

$$\frac{1}{\gamma_N(T)} - 1 \approx 1 - \gamma_N(T). \quad (21)$$

The resulting algorithm is presented in Table 2. Notice that the only additional operations that will increase the complexity are those required for the computation of the backward residual as defined in (8) (N multiplications and additions). Also in Table 2, the variable $k_N(T-1)$ is computed as $\lambda^{-N}\gamma_N(T-1)A^N_{N,T-1}$ instead of $\beta_N(T-1)A^N_{N,T-1}/\alpha_N(T-1)$ as was initially defined. The two expressions are equivalent under infinite precision but the first avoids one division. The only additional divisions are those required for the computation of $\xi_N(T)$ and $\bar{\xi}_N(T)$.

Comments. A parameter of our algorithm that needs to be defined is the constant ρ. The algorithm is quite robust regarding this parameter, although very small or very large values will make it unstable. We will give a means for estimating a value for ρ. Substituting in $w_N(T)$ defined in (13) the optimal values for the corrections from (14) and (15), yields

$$\begin{aligned}\min w_N(T) &= \rho^2[\bar{\xi}_N(T)]^2(k_N(T-1))^2\left[\frac{1}{\gamma_N(T-1)} - 1\right] \\ &\quad + \rho^2[\bar{\xi}_N(T)]^2\left[\frac{1}{\gamma_N(T)} - 1\right] + \rho[\bar{\xi}_N(T)]^2 \\ &\approx \rho^2[\bar{\xi}_N(T)]^2\left[\frac{1}{\gamma_N(T)} - 1\right] + \rho[\bar{\xi}_N(T)]^2. \end{aligned} \quad (22)$$

The approximation is usually valid since for most signals we have $|A^N_{N,T}| \ll 1$ and thus we will also have $|k_N(T-1)| \ll 1$. The ratio of the two terms in (22) is $\rho(1/\gamma_N(T) - 1)$. Approximating $\gamma_N(T)$ with λ we have that the ratio of the two terms is $\rho(1/\lambda - 1)$. From simulations it was observed that our algorithm worked best when the ratio had values around 0.05. Thus, a first estimation for ρ can be

$$\rho = \rho_0\frac{\lambda}{1-\lambda} \quad \text{with} \quad \rho_0 \approx 0.05. \quad (23)$$

From (23) we can see that ρ is basically related to λ and not to the order N. A correct selection of λ depends heavily on the eigenvalue spread of the matrix $R_{N,T}$ (defined by (4)). Details on this dependence and optimal values for λ can be found in [10]. Since usually λ overestimates $\gamma_N(T)$, we have that (23) overestimates ρ. As one can see this overestimation becomes more severe for large λ close to unity. Thus (24) is used, merely to indicate the order of magnitude of ρ and not ρ itself.

It is possible to simplify even further our algorithm. Notice from Table 2 that in the corrections there is always the term $\rho\bar{\xi}_N(T)$. Using (17, 22), neglecting the term with $k_N(T-1)$ and approximating $\gamma_N(T)$ with λ we have

$$\rho\bar{\xi}_N(T) \approx \frac{\rho}{1+\rho(1-\lambda)}\xi_N(T). \quad (24)$$

Table 2
Stabilized fast a-posteriori error sequential technique (SFAEST)—complexity $8N$

	Multiplications	Divisions
Available at time T: $W_{N,T-1}, A_{N,T-1}, B_{N,T-1}, H_{N,T-1}, X_{N,T-1}, \gamma_N(T-1), \alpha_N(T-1), \beta_N(T-1)$.		
New information: $y(T), x(T)$.		
Computation of the residuals and their corrections		
$e_N^f(T) = x(T) - A'_{N,T-1} X_{N,T-1}$	N	—
$e_N^b(T) = x(T-N) - B'_{N,T-1} X_{N,T}$	N	—
$\gamma_{N+1}(T) = \dfrac{\lambda\alpha_N(T-1)}{\lambda\alpha_N(T-1) + \gamma_N(T-1)(e_N^f(T))^2}\gamma_N(T-1)$	4	1
$\theta_N(T) = 1 + \gamma_{N+1}(T) e_N^b(T)\left(W_{N,T-1}^N + \dfrac{e_N^f(T)}{\lambda\alpha_N(T-1)} A_{N,T-1}^N\right)$	3	1
$\gamma_N(T) = \dfrac{\gamma_{N+1}(T)}{\theta_N(T)}$	—	1
$k_N(T-1) = \lambda^{-N}\gamma_N(t-1) A_{N,T-1}^N$	2	—
$\xi_N(T) = e_N^b(T) + k_N(T-1) e_N^f(T) + \lambda\beta_N(T-1) W_{N,T-1}^N$	3	—
$\bar{\xi}_N(T) = \{1 + \rho(1-\gamma_N(T)) + \rho(k_N(T-1))^2(1-\gamma_N(T-1))\}^{-1}\xi_N(T)$	4	1
$\bar{e}_N^f(T) = e_N^f(T) - (1-\gamma_N(T-1)) k_N(T-1)\rho\bar{\xi}_N(T)$	3	—
$\alpha_N(T) = \lambda\alpha_N(T-1) + \gamma_N(T-1)(\bar{e}_N^f(T))^2$	3	—
$\bar{e}_N^b(T) = e_N^b(T) - (1-\gamma_N(T))\rho\bar{\xi}_N(T)$	2	—
$\beta_N(T) = \lambda\beta_N(T-1) + \gamma_N(T)(\bar{e}_N^b(T))^2$	3	—
Time update of the Kalman Gain $W_{N,T}$		
$W_{N+1,T} = \begin{bmatrix} 0 \\ W_{N,T-1} \end{bmatrix} - \dfrac{\bar{e}_N^f(T)}{\lambda\alpha_N(T-1)}\begin{bmatrix} 1 \\ -A_{N,T-1} \end{bmatrix}$	$N+1$	1
$A_{N,T} = A_{N,T-1} - \gamma_N(T-1)(e_N^f(T) + k_N(T-1)\rho\bar{\xi}_N(T)) W_{N,T-1}$	$N+4$	—
$\begin{bmatrix} W_{N,T} \\ 0 \end{bmatrix} = W_{N+1,T} - W_{N+1,T}^{N+1}\begin{bmatrix} -B_{N,T-1} \\ 1 \end{bmatrix}$	N	—
$B_{N,T} = B_{N,T-1} - \gamma_N(T)(e_N^b(T) + \rho\bar{\xi}_N(T)) W_{N,T}$	$N+3$	—
Time update of the filter $H_{N,T}$		
$e_N(T) = y(T) - H'_{N,T} X_{N,T}$	N	—
$\varepsilon_N(T) = \gamma_N(T) e_N(T)$	1	—
$H_{N,T} = H_{N,T-1} - \varepsilon_N(T) W_{N,T}$	N	—
Total number of multiplications/divisions	$8N+36$	5

With this simplification we avoid one more division and certain multiplications since the term in front of $\xi_N(T)$ can be precomputed.

Even though our algorithm seems to work at least for a number of data of several hundred thousands, there is always the possibility of a very slow instability. There is thus the need of a detection procedure. In [8] the variable $\gamma_N(T)$ of the algorithm is monitored and the algorithm is reinitialized when this variable is outside the interval [0 1]. As stated in [3], this detection procedure works for the case where we have divergence

toward infinity. There is unfortunately another mode of divergence [2], which is less known and difficult to observe. It consists of a gain tending to zero (and thus $\gamma_N(T)$ tends to zero as well) with filters having acceptable values. In other words the tracking capability of the algorithm becomes small. This is very severe for the nonstationary data sequence case. In our opinion the following rule does not seem to have the problems of $\gamma_N(T)$: reinitialize when $\xi_N^2(T) > t\beta_N(T)$ where t is a small constant. For both modes of divergence $\xi_N(T)$ increases in absolute value. Thus this rule will detect instability efficiently in both cases.

3. Simulations

We present the case $N=20$ for $\{x(T)\}$ being a white Gaussian noise sequence with zero mean and unit variance. The parameters of the algorithm are the following: $\lambda = 0.98$, $\rho = 2$. We initialize the algorithm with $\alpha_N(0) = \mu\lambda^N$, $\beta_N(0) = \mu$, $\gamma_N(0) = 1$ and $\mu = 0.1$ (similar example as in [7]). The simulations were performed with a PC and with 32 bits floating point arithmetic. In Fig. 1 we plot $\gamma_N(T)$ and $\xi_N(T)$ for the algorithm of Table 1 but with the backward residual computed as in (8). This algorithm has complexity $8N$, comparable to our algorithm, but it is more stable than the $7N$ FAEST which is highly unstable. When the algorithm works correctly then $0 \leq \gamma_N(T) \leq 1$. The algorithm is reinitialized every time we have $\gamma_N(T) > 1$ or $\gamma_N(T) < 0$. We can also see the behavior of the variable $\xi_N(T)$; its magnitude gradually increases with time as the round-off errors accumulate. Notice that we performed a 10 000 points simulation. Fig. 2 has the performance of the stabilized FAEST (SFAEST) algorithm of Table 2 for the same input sequence. We can see that $\gamma_N(T)$ has always a value less than unity and $\xi_N(T)$ is very small. For this case we performed a 100 000 points simulation. The periodic nature of the graph is due to the recycling of a data sequence of length 10 000. The recycling does not favor the stability of the

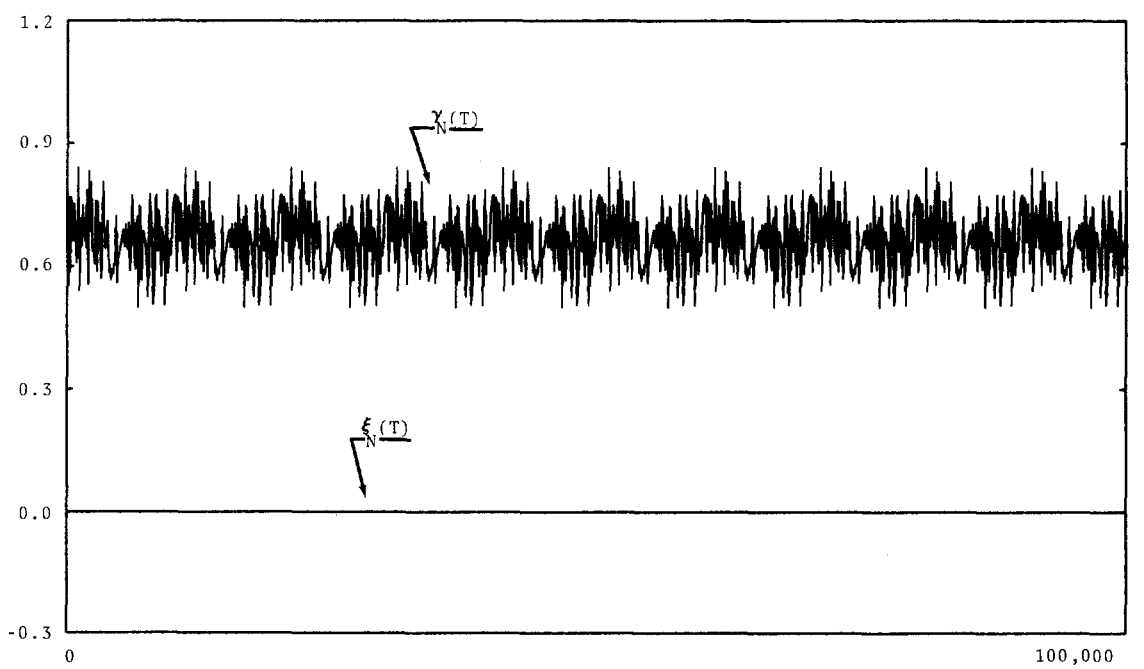

Fig. 2. Performance of the SFAEST algorithm for $N=20$ and for white noise data sequence.

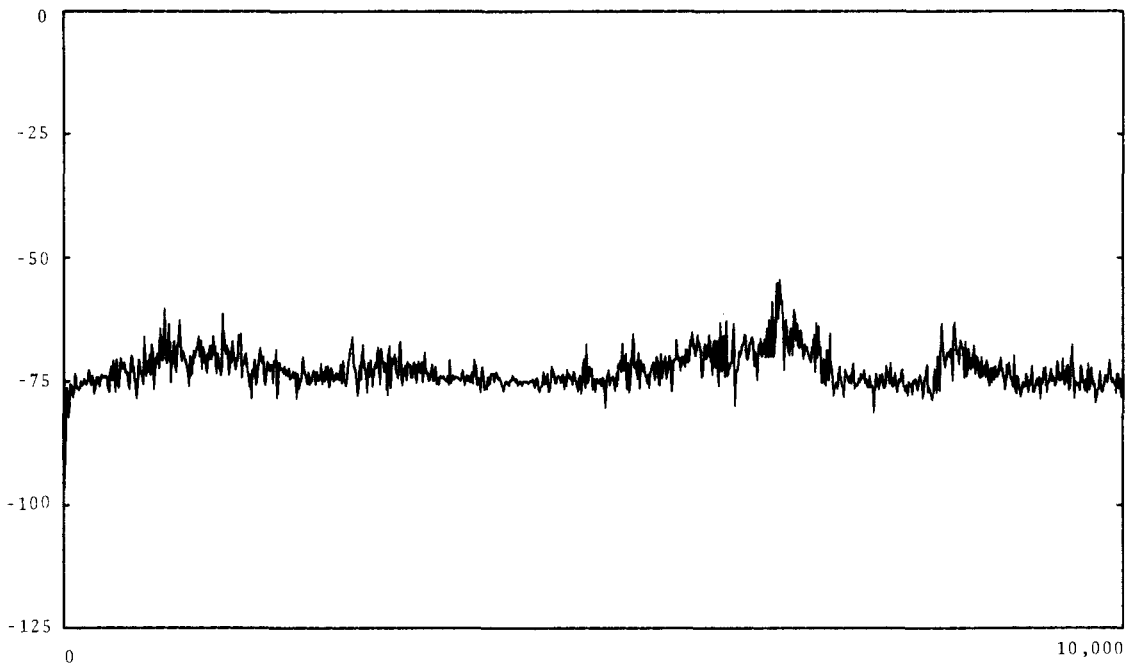

Fig. 3. Relative difference (in dB) of the Kalman Gains computed by RLS and SFAEST.

algorithm because instability is basically a feature of the algorithm itself and it is present even when the data are well behaved. As in [3] we will compare the Dual Kalman Gains of the SFAEST and the RLS to see if the proposed algorithm is still a realization of the least-squares. We use the RLS Algorithm for comparisons, because when it is performed in the most non-fast way, it is a very stable realization of least-squares. Thus, in Fig. 3, we plot $\|W_{RLS} - W_{SFAEST}\|/\|W_{RLS}\|$ (in dB); the relative difference of the two Dual Kalman Gains. We can see that the two gains are very close to each other.

4. Conclusions

In this paper we derived a stabilized version of the FAEST algorithm. The idea used to stabilize the algorithm was similar to a previous work, but our derivation was easier and led to an algorithm of significantly lower complexity. We do not claim that the method introduced here can stabilize all FKA versions in all situations. As it is stated in [3], if the forgetting factor is not compatible with the data, then no least-squares version is stable. We believe though, that for most cases where the forgetting factor λ and the parameter ρ are correctly selected, our method will stabilize or at least will increase the stable life of the corresponding FKA by a significant factor.

References

[1] M. Bellanger, "Engineering aspects of fast least squares algorithms in transversal adaptive filters", *Proc. Int. Conf. Acoust. Speech Signal Process.*, Dallas, Paper No. 49.14, 1987.

[2] J.L. Botto, "Etude des algorithmes transversaux rapides: application a l'annulation d'echo acoustiques pour l'audioconférence", *These de Docteur Ingénieur*, Université de Rennes I, IRISA, France, May 1986.

[3] J.L. Botto and G.V. Moustakides, "Stabilization of fast recursive least-squares transversal filters", *INRIA-IRISA Rapports de Recherche*, No. 570, September 1986.

[4] G. Carayannis, D. Manolakis and N. Kalouptsidis, "A fast sequential algorithm for least-squares filtering and prediction", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 31, No. 6, December 1983, pp. 1394-1402.

[5] G. Carayannis, D. Manolakis and N. Kalouptsidis, "Fast Kalman type algorithms for sequential signal processing", *Proc. Int. Conf. Acoust. Speech, Signal Process.*, Boston, 1983, pp. 186-189.

[6] G. Carayannis, D. Manolakis and N. Kalouptsidis, "A unified view of parametric processing algorithms for prewindowed signals", *Signal Process.*, Vol. 10, No. 4, June 1986, pp. 335-368.

[7] J.M. Cioffi, "Limited precision effects in adaptive filtering", *IEEE Trans. Circuits Syst.* (Special Issue on Adaptive Filtering), Vol. CAS-34, July 1987, pp. 1097-1110.

[8] J.M. Cioffi and T. Kailath, "Fast recursive least-squares transversal filters for adaptive filtering", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 32, No. 2, April 1984, pp. 304-337.

[9] J.M. Cioffi and T. Kailath, "Windowed fast transversal filters adaptive algorithms with normalization", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 33, No. 3, June 1985, pp. 607-625.

[10] E. Eleftheriou and D.D. Falconer, "Tracking properties and steady-state performance of RLS adaptive filter algorithms", *IEEE Trans. Acoust. Speech Signal Process*, Vol. 34, No. 5, October 1986, pp. 1097-1110.

[11] P. Fabre and C. Gueguen, "Fast recursive least-squares algorithms: preventing divergence", *Proc. Int. Conf. Acoust. Speech Signal Process.*, Tampa, Paper No. 30.2, March 1985.

[12] P. Fabre and C. Gueguen, "Improvement of fast RLS algorithms via normalization: a comparative study", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 34, No. 2, April 1986, pp. 296-308.

[13] D.D. Falconer and L. Ljung, "Application of fast Kalman estimation to adaptive equalization", *IEEE Trans. Commun.*, Vol. COM-26, October 1987, pp. 1439-1446.

[14] L. Ljung, M. Morf and D. Falconer, "Fast calculation of gain matrices for recursive estimation schemes", *Int. J. Control*, Vol. 27, January 1978, pp. 1-19.

[15] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*, MIT Press, Cambridge, MA, 1983.

[16] S. Ljung and L. Ljung, "Error Propagation Properties of Recursive Least-Squares Adaptation Algorithms", *Automatica*, Vol. 21, No. 2, 1985, pp. 157-167.

[17] D. Manolakis, G. Carayannis and N. Kalouptsidis, "New issues on the computational organization of fast sequential algorithms", *Proc. Int. Conf. Acoust. Speech Signal Process.*, San Diego, Paper No. 43.7, 1984.

[18] A. Nehorai and M. Morf, "A unified derivation for fast estimation algorithms by the conjugate direction method", *Lin. Alg. Appl.*, Vol. 72, 1985, pp. 119-143.