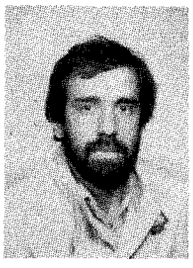
Implantation en temps réel d'un algorithme de moindres carrés rapide sur microprocesseur de traitement du signal

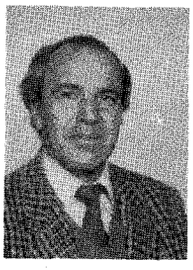
Implementation in real time of a fast least squares on a Digital Signal Processor



J. M. TRAVASSOS-ROMANO

Laboratoire des Signaux et Systèmes, CNRS-ESE, Plateau du Moulon, 91192 GIF-SUR-YVETTE.

Ingénieur de l'Université de Campinas (UNICAMP, Brésil 1981), Master of Sciences (1984), Docteur de l'Université Paris-Sud (1987). Il a fait ses travaux de recherche au Laboratoire des Signaux et Systèmes. Domaines d'intérêt : Filtrage numérique, filtrage adaptatif, transmission numérique, analyse des signaux. Il enseigne actuellement à UNICAMP.



J. STRUB

Laboratoire des Signaux et Systèmes, CNRS-ESE, Plateau du Moulon, 91192 GIF-SUR-YVETTE.

Ingénieur d'Études au CNRS, Travaille au Laboratoire des Signaux et Systèmes depuis sa création. Domaines d'intérêt : filtrage numérique, filtrage adaptatif. Développe actuellement un logiciel de traitement des signaux électroencéphalographiques.

RÉSUMÉ

Les résultats d'une implantation en temps réel d'un algorithme de moindres carrés rapide (MCR) sont présentés et étudiés dans cet article, dans le cas particulier de la prédiction linéaire de signaux sinusoïdaux bruités.

Un algorithme MCR présente fondamentalement deux modes de divergence. Nous proposons deux méthodes simples pour le rendre stable à long terme et nous étudions l'effet de ces méthodes sur les performances de l'algorithme.

Enfin, la méthodologie suivie dans l'expérience est discutée et les résultats sont présentés sous forme de courbes obtenues sur l'oscilloscope.

MOTS CLÉS

Filtre numérique adaptatif, prédiction linéaire, moindres carrés rapide, stabilisation, implantation, temps réel.

SUMMARY

This paper presents a real-time implementation of a fast least-squares algorithm (FLS), for the linear prediction of sinusoidal signals in noise, and analyzes the results.

The FLS algorithm has two divergence modes. We propose two simple techniques which provide long-term stability, and study their effects on the performance of the algorithm.

The experimental method is discussed and results are presented in the form of oscilloscope curves.

KEY WORDS

Adaptive digital filter, linear prediction, fast least-squares, stabilization, implementation, real-time.

1. Introduction

Le filtrage adaptatif est devenu, depuis ces deux dernières décades, une technique de base pour l'analyse de signaux en temps réel [2]. Le principe d'un filtre numérique adaptatif consiste à mettre en œuvre un algorithme qui calcule, à chaque nouvelle valeur de l'entrée et de l'erreur en sortie $e(n)$, les coefficients du filtre selon un critère pré-établi.

Le critère le plus souvent utilisé consiste à minimiser une moyenne sur l'erreur quadratique, donnée par :

$$J_1(n) = E[e^2(n)] \tag{1}$$

où $E[.]$ représente l'espérance mathématique d'une variable aléatoire.

Ce critère est à l'origine des techniques du gradient. Dans un contexte ergodique il correspond, en régime permanent, à une moyenne temporelle qui est le critère établi pour les algorithmes des moindres carrés. Pour leur donner une capacité de poursuite, le critère est généralisé par l'introduction d'un facteur d'oubli w et la fonction coût à minimiser est :

$$J_2(n) = \sum_{p=1}^{n} w^{n-p} e^2(p). \tag{2}$$

Les techniques des moindres carrés ont des avantages importants par rapport à celles du gradient [1], d'où l'intérêt des algorithmes des moindres carrés rapides (MCR), qui réalisent le critère J_2 avec une complexité des calculs proportionnelle au nombre des coefficients du filtre.

Néanmoins, les algorithmes MCR associés à une structure de filtre transversale présentent un comportement souvent instable, attribué en général à l'accumulation des erreurs d'arrondi, inévitables à cause de la précision finie des machines employées dans sa mise en œuvre.

Il existe, dans la littérature, plusieurs études importantes sur la stabilisation des algorithmes MCR, mais l'absence de résultats obtenus sur des signaux réels provoque encore une réticence de la part des utilisateurs et empêche que cet algorithme soit plus répandu en pratique, malgré tout son potentiel.

Le but de cet article est de montrer que ce type d'algorithme, convenablement adapté, peut fonctionner avec des signaux réels, de manière continue, et de présenter des résultats pour le cas de la prédiction linéaire en temps réel de sinusoïdes bruitées. Après avoir fait un rappel sur les principes de l'algorithme, nous proposons une méthode simple pour le rendre stable à long terme et nous étudions les effets de cette méthode sur les performances de l'algorithme. Ensuite, nous décrivons le montage expérimental et la méthodologie d'implantation que nous avons suivie. Les résultats obtenus en temps réel sont présentés à la fin et illustrés par des courbes sur l'oscilloscope.

2. Filtrage et prédiction linéaire avec les algorithmes MCR

Le schéma d'un filtre adaptatif est représenté sur la figure 1. Si $H(n)$ est le vecteur des N coefficients $h_i(n)$, $0 \leqq i \leqq N-1$, et $X(n)$ le vecteur des N dernières

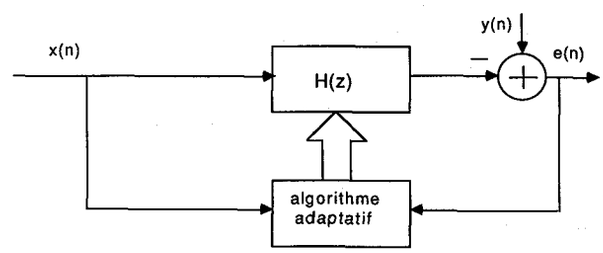


Fig. 1. – Schéma du filtre adaptatif.

valeurs reçues du signal d'entrée $x(n)$, alors l'erreur en sortie est donnée par :

$$e(n) = y(n) - H^t(n) X(n) \tag{3}$$

où $y(n)$ est un signal de référence.

Les coefficients optimaux $H(n)$ sont ceux qui minimisent le critère donné en (2) où l'on suppose nul $x(p)$ pour $p < 1$.

$$J_2(n) = \sum_{p=1}^{n} w^{n-p} \left| y(p) - H^t(n) X(p) \right|^2 \tag{4}$$

La solution, à l'instant n, est l'ensemble des N coefficients donnés par les équations normales

$$H(n) = R_N^{-1}(n) R_{yx}(n) \tag{5}$$

où $R_N(n)$ est une estimation de la matrice d'autocorrélation du signal d'entrée $x(n)$ et $R_{yx}(n)$ le vecteur

des coefficients d'intercorrélation entre les signaux d'entrée et de référence.

$$(6) \qquad R_N(n) = \sum_{p=1}^{n} w^{n-p} X(p) X^t(p)$$

$$(7) \qquad R_{yx}(n) = \sum_{p=1}^{n} w^{n-p} X(p) y(p)$$

L'équation (5) a aussi une solution par récurrence. Les coefficients sont alors, à chaque itération, obtenus par :

$$(8) \quad H(n) = H(n-1) + R_N^{-1}(n) X(n)] \times [y(n) - H^t(n-1) X(n)].$$

L'adaptation des coefficients $H(n)$ par l'équation (8) demande la mise à jour, à chaque instant n, de la matrice $R_N^{-1}(n)$. Cette opération présente une quantité de calculs proportionnelle au carré du nombre N de coefficients du filtre et, pour des fréquences d'échantillonnage élevées, les machines correspondantes peuvent être d'un coût prohibitif dès que N dépasse quelques unités.

L'intérêt des algorithmes de moindres carrés rapides est de fournir, à chaque instant n, les mêmes coefficients optimaux donnés par (8) avec une quantité de calculs seulement proportionnelle à N. Le principe consiste à mettre à jour un gain d'adaptation défini par :

$$(9) \qquad G(n) = R_N^{-1}(n) X(n)$$

Toute l'information nécessaire à l'obtention de ce gain est en fait contenue dans l'ensemble des coefficients et des erreurs de prédiction linéaire. Il convient donc de faire un rappel sur le calcul de ces paramètres.

Le filtre de prédiction linéaire correspond au cas particulier où le signal de référence est le signal d'entrée lui-même. Les coefficients de prédiction constituent l'ensemble optimal qui minimise le critère des moindres carrés suivant :

$$J_3(n) = \sum_{p=1}^{n} w^{n-p} [x(p) - A^t(n) X(p-1)]^2$$

l'erreur de prédiction en sortie dite *a priori* est définie par :

$$(10) \qquad e_a(n+1) = x(n+1) - A^t(n) X(n)$$

et les cœfficients s'expriment en fonction du signal d'entrée par :

$$(11) \qquad A(n) = R_N^{-1}(n-1) r_N^a(n)$$

avec :

$$A^t(n) = [a_1(n)\, a_2(n) \ldots a_N(n)]$$

$$r_N^a(n) = \sum_{p=1}^{n} w^{n-p} X(p-1) x(p).$$

De même que l'apport dû à une nouvelle donnée d'entrée $x(n+1)$ peut être pris en compte par la relation (10), on peut établir une relation similaire correspondant à l'abandon d'information dû à la sortie de $x(n-N+1)$. Il est donc possible de caractériser le signal par un autre type de filtre, orienté dans le sens des indices décroissants, appelé filtre de prédiction rétrograde ou prédiction arrière, dont l'erreur en sortie a priori est définie par :

$$(12) \quad e_b(n+1) = x(n-N+1) - \sum_{i=1}^{N} b_i(n) x(n-N+1+i).$$

Le vecteur $B(n)$, de l'ensemble des coefficients arrière $b_i(n)$, est aussi obtenu selon un critère de moindres carrés qui consiste à minimiser la quantité :

$$J_4(n) = \sum_{p=1}^{n} w^{n-p} [x(p-N) - B^t(n) X(p)]^2.$$

Dans ces conditions les coefficients sont donnés par :

$$(13) \qquad B(n) = R_N^{-1}(n) r_N^b(n)$$

où

$$r_N^b(n) = \sum_{p=1}^{n} w^{n-p} X(p) x(p-N).$$

A partir du gain d'adaptation défini dans l'expression (9), il est possible de trouver une solution de récurrence pour les coefficients avant et arrière :

$$(14) \qquad A(n+1) = A(n) + G(n) e_a(n+1)$$

$$(15) \qquad B(n+1) = B(n) + G(n+1) e_b(n+1).$$

En introduisant les coefficients déjà mis à jour dans les expressions (10) et (12), on obtient les erreurs dites *a posteriori*, données par :

$$(16) \qquad \varepsilon_a(n+1) = x(n+1) - A^t(n+1) X(n)$$

$$(17) \quad \varepsilon_b(n+1) = x(n-N+1) - B^t(n+1) X(n+1).$$

Les algorithmes MCR peuvent être classés dans deux groupes, les algorithmes basés sur l'erreur *a priori*, parfois appelés algorithmes de Kalman rapide, qui se caractérisent par le calcul récurrent du gain d'adaptation [3]. Le deuxième groupe comprend les algorithmes basés sur l'ensemble des erreurs de prédiction, ils utilisent les erreurs *a posteriori* pour l'adaptation des coefficients et calculent un gain dual, donné par [4] :

$$(18) \qquad G'(n) = R_N^{-1}(n-1) X(n).$$

Nous avons implanté sur microprocesseur l'algorithme du premier groupe, dont l'ensemble des opérations est décrit dans le tableau de la figure 2. Les principaux problèmes d'implantation, tels que le cadrage et la stabilisation, sont communs à tous les algorithmes et nos conclusions peuvent être étendues aux autres. On trouve dans la référence [5] une étude complète sur l'implantation en arithmétique entière de plusieurs types d'algorithmes MCR.

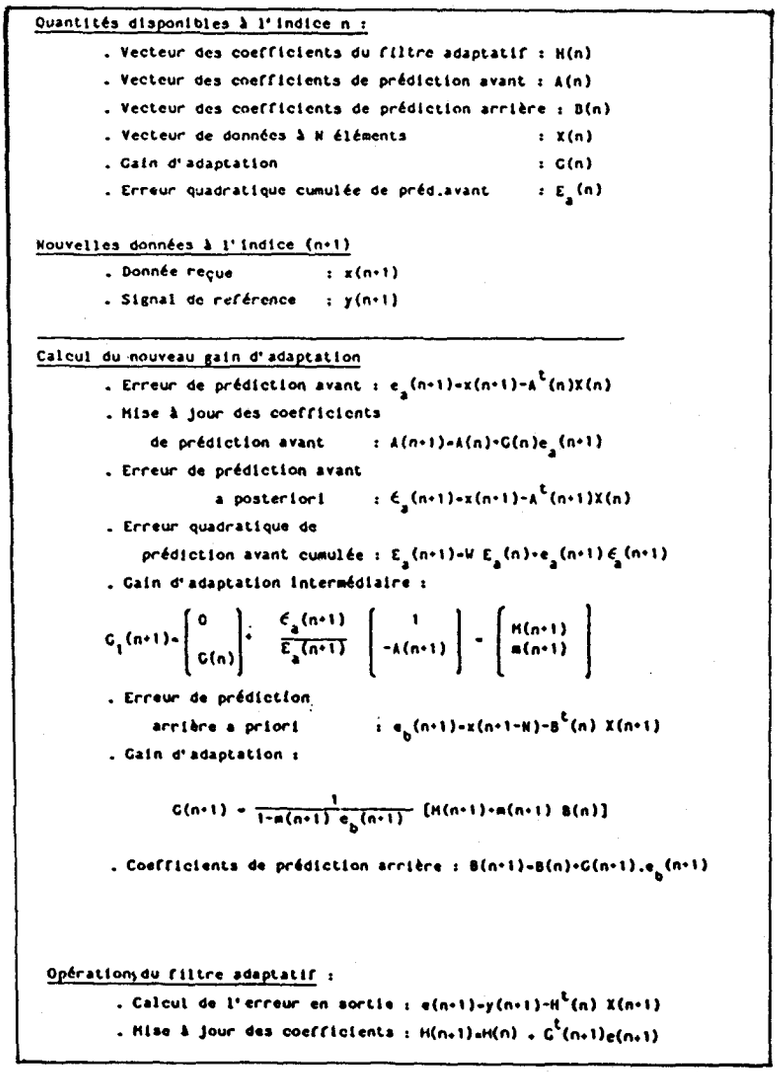


Fig. 2. — Algorithme MCR.

3. Conditions de stabilité

Nous avons restreint notre étude au cas de la prédiction linéaire. Dans ce cas les deux dernières opérations de la figure 2 ne sont pas réalisées, car les coefficients de prédiction sont directement fournis par le vecteur $A(n)$.

Il existe fondamentalement deux modes de divergence dans les algorithmes MCR. Nous allons les présenter séparément et montrer ensuite les résultats obtenus avec les méthodes de stabilisation appliquées.

3.1. La constante de stabilisation

Si le spectre d'un signal appliqué à l'entrée d'un filtre de prédiction est composé de raies, l'erreur en sortie sera nulle quand les valeurs optimales des coefficients auront été atteintes. Le premier mode de divergence de l'algorithme apparaît alors car la variable exprimant l'énergie de l'erreur et calculée par :

$$(19)\qquad E_a(n+1)=w\,E_a(n)+e_a(n+1)\,\varepsilon_a(n+1),$$

tendra vers zéro si $w<1$. Ceci est inacceptable à cause de la division effectuée dans les calculs du gain intermédiaire $G_1(n+1)$. Par contre, un facteur d'oubli $w<1$ est nécessaire pour que l'algorithme puisse suivre les évolutions du signal d'entrée. En pratique, l'erreur cumulée $E_a(n)$ doit être toujours supérieure à un certain seuil, de façon à ce que la dynamique des gains soit bornée dans des limites raisonnables, en fonction de la précision de la machine. Ce résultat peut être obtenu avec l'introduction d'une constante, dite de stabilisation, dans l'expression (19) :

$$(20)\qquad E_a(n+1)=w\,E_a(n)+e_a(n+1)\,\varepsilon_a(n+1)+\mathscr{C}.$$

Il vient alors :

$$(21)\qquad E_a(n)\geqq\frac{\mathscr{C}}{1-w}.$$

L'introduction de cette constante provoque, évidemment, un écart par rapport au critère rigoureux des moindres carrés et la stabilité de principe, liée à ce

critère, ne peut plus être garantie. Cette perturbation est plus importante quand il s'agit d'un signal plus difficile à prédire et où la moindre modification par rapport au critère rigoureux peut rendre impossible la convergence vers les coefficients optimaux.

Une illustration de ce comportement est faite avec le cas simple d'un prédicteur d'ordre 2 appliqué à une sinusoïde. La prédiction est d'autant plus difficile que la fréquence de la sinusoïde est proche de zéro ou de la demi-fréquence d'échantillonnage ($f_e/2$). Par rapport à une fréquence placée, dans le plan des Z, plus loin de l'axe réel, cette difficulté de prédiction se traduit par deux faits :

– Les valeurs des coefficients optimaux, et en conséquence l'estimation de la fréquence, sont plus sensibles au bruit ajouté à la sinusoïde.

– La convergence de l'algorithme est plus lente.

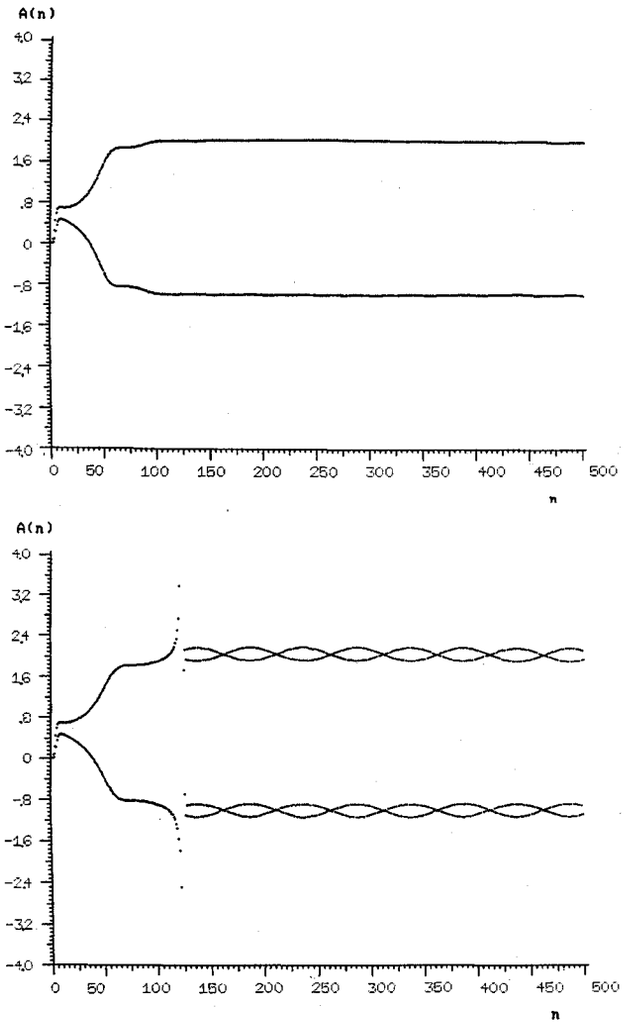


Fig. 3. – Évolution des coefficients de prédiction pour $f=f_e/100$.
(A) $\mathscr{C}=0$.
(B) $\mathscr{C}=E_a(0)/100$.

La figure 3 montre la convergence de l'algorithme MCR pour une fréquence de 100 Hz (échantillonnage à 10 kHz), facteur d'oubli $w=0{,}9375$ et $E_a(0)=0{,}125$ comme énergie initiale d'erreur. Dans le premier cas (*fig.* 3 A), il n'y a pas de constante de stabilisation et, dans le cas de la figure 3 B, elle vaut $E_a(0)/100$. On constate donc l'existence d'une divergence, après laquelle les coefficients trouvent une situation d'équilibre autour des valeurs optimales.

Dans la figure 4, l'expérience est renouvelée avec une sinusoïde à 2 kHz et on remarque que la constante de stabilisation a un effet négligeable sur la courbe de convergence. Il convient de rappeler que ces simulations ont été faites pour 1 000 itérations, en laissant tourner l'algorithme pendant un temps plus long, une divergence apparaît à cause de l'annulation de l'erreur cumulée lorsque $\mathscr{C}=0$.

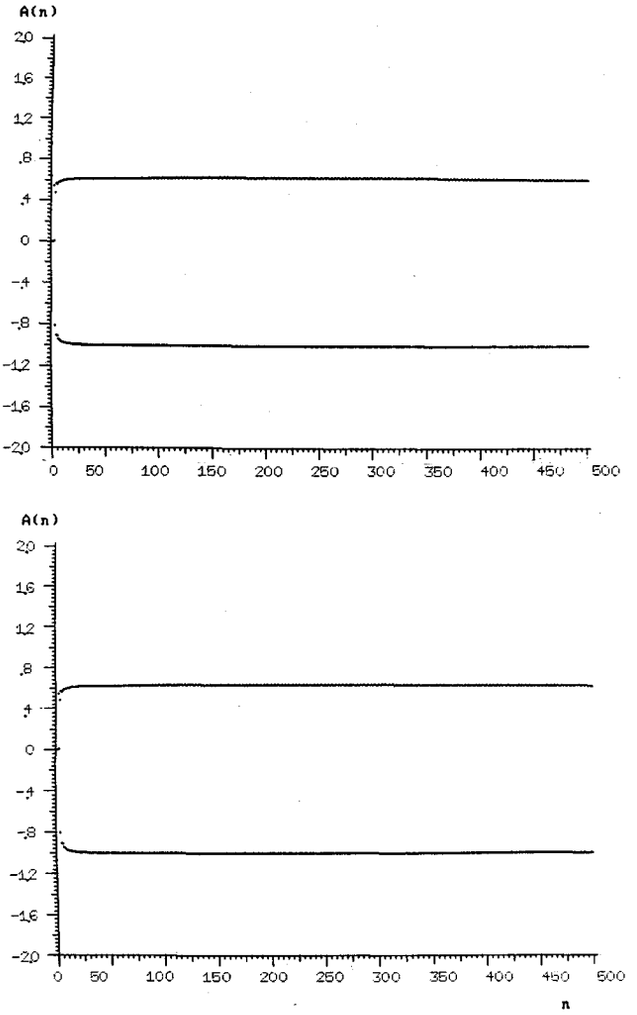


Fig. 4. – Évolution des coefficients de prédiction pour $f=f_e/5$.
(A) $C=0$.
(B) $C=E_a(0)/100$.

La stabilité de l'algorithme est directement liée au comportement de la variable qui exprime le rapport entre les erreurs de prédiction *a posteriori* et *a priori* :

$$\varphi(n)=\frac{\varepsilon_a(n+1)}{e_a(n+1)}. \tag{22}$$

Cette variable doit toujours être comprise entre 0 et 1 si l'algorithme est stable. En fait, on peut montrer que :

$$\varphi(n)=1-X^t(n)\,R_N^{-1}(n)\,X(n) \tag{23}$$

où $R_N(n)$ est définie par (6). A la convergence, cette matrice devient pratiquement constante et, si le facteur d'oubli est proche de 1, elle est approximativement donnée par [1] :

$$R_N^{-1}(n) \cong (1-w) R_{xx}^{-1} \tag{24}$$

où R_{xx} est la vraie matrice d'autocorrélation du signal d'entrée.

Pour une entrée $x(n) = A \cdot \sin(n\omega)$ et un prédicteur d'ordre 2, il vient :

$$R_{xx}^{-1} = \begin{bmatrix} \frac{1}{\sin^2\omega} & -\frac{\cos\omega}{\sin^2\omega} \\ -\frac{\cos\omega}{\sin^2\omega} & \frac{1}{\sin^2\omega} \end{bmatrix} x \frac{2}{A^2} \tag{25}$$

et

$$\varphi(n) = 1-(1-w)\, A \begin{bmatrix} \sin(n\omega) \\ \sin((n-1)\omega) \end{bmatrix}^t \times R_{xx}^{-1} A \begin{bmatrix} \sin(n\omega) \\ \sin((n-1)\omega) \end{bmatrix} \tag{26}$$

En effectuant les calculs, il vient :

$$\varphi(n) \cong 2w-1 \tag{27}$$

pour les grandes valeurs de l'indice n.

Le tableau ci-dessous, obtenu par simulation dans le cas de la prédiction d'une sinusoïde avec $\omega = 2\pi/5$, montre que l'expression (27) est d'autant plus vraie que w est proche de 1.

Oubli w	φ à la convergence	$(2w-1)$
0,99	0,98	0,98
0,98	0,96	0,96
0,95	0,90	0,90
0,9	0,81	0,80
0,85	0,72	0,70
0,8	0,64	0,60

Le comportement de l'algorithme en fonction de la fréquence de la sinusoïde à l'entrée du prédicteur peut être analysé si l'on considère, à partir des expressions (19) et (24), que le vecteur gain à la convergence est donné par :

$$G(n) = (1-w) R_{xx}^{-1} X(n) = \begin{bmatrix} g_1(n) \\ g_2(n) \end{bmatrix} \tag{28}$$

En effectuant les calculs, pour $x(n) = A\sin(n\omega)$, il vient :

$$g_1(n) = \frac{2}{A} \frac{\cos[(n-1)\omega]}{\sin\omega} (1-w) \tag{29}$$

$$g_2(n) = \frac{-2}{A} \frac{\cos(n\omega)}{\sin\omega} (1-w) \tag{30}$$

Alors, les gains oscillent avec la fréquence du signal d'entrée et, si $\sin\omega$ devient petit, ils peuvent atteindre des valeurs très élevées. Il est donc clair que, dans une réalisation pratique où les gains doivent être bornés par des valeurs limites, il y a une plage de fréquence en dehors de laquelle l'algorithme n'est pas stable, à cause des saturations des variables.

3.2. Rappel à zéro sur les coefficients

Le deuxième mode de divergence des algorithmes MCR en structure transversale est dû à l'accumulation des erreurs d'arrondis dans les machines à précision finie. Cette divergence se traduit par une dérive des valeurs des coefficients de prédiction [6]. Une technique simple pour compenser cette dérive consiste à introduire sur les coefficients de prédiction un rappel à zéro, tel que :

$$\begin{cases} A'(n) = (1-\delta) A(n) \\ B'(n) = (1-\delta) B(n), \end{cases} \quad \text{avec} \quad 0 < \delta \ll 1 \tag{31}$$

On peut montrer que la constante de stabilisation introduite précédemment contribue, pour certains types de signaux, à effectuer ce rappel à zéro.

Ces opérations provoquent un biais sur les valeurs des coefficients qui peut être estimé si l'on considère que, en régime permanent, les algorithmes MCR ont un comportement équivalent au gradient avec un pas d'adaptation $(1-w)$ [1]. Alors, les valeurs des coefficients de prédiction après convergence sont données d'après [2, chap. 4], par :

$$A(n) = \left[\frac{\delta}{1-w} I + R_{xx} \right]^{-1} r_N^a(n), \qquad n \to \infty \tag{32}$$

L'utilisateur doit alors choisir un facteur de rappel à zéro raisonnable, de façon à compenser les erreurs d'arrondi sans introduire un biais trop fort sur les coefficients. En pratique, nous avons utilisé $\delta \ll 1-w$.

Il faut noter que le rappel à zéro des coefficients est intéressant pour le comportement de l'algorithme quand les signaux à l'entrée sont entrecoupés de séquences nulles. Les résultats obtenus expérimentalement, présentés et analysés dans la suite, confirment l'importance de cette technique pour la stabilisation de l'algorithme et pour son bon fonctionnement après des coupures sur le signal d'entrée.

4. Réalisation en temps réel

La réalisation d'un algorithme en temps réel nécessite de disposer de sa formulation, d'un matériel capable de le mettre en œuvre et d'une traduction, sous forme d'instructions logiques, qui permette l'exécution des opérations correspondant à la formulation établie.

Dans un premier temps, nous allons décrire le système qui comprend le matériel et le logiciel employés pour passer ensuite à la méthodologie d'implantation où le problème de la mise en échelle est analysé.

4.1. Description du banc d'essai

La figure 5 montre le schéma du banc d'essai dont les principaux éléments sont :

– Une carte d'évaluation EVM qui comprend un microprocesseur TMS 9995 avec son moniteur en

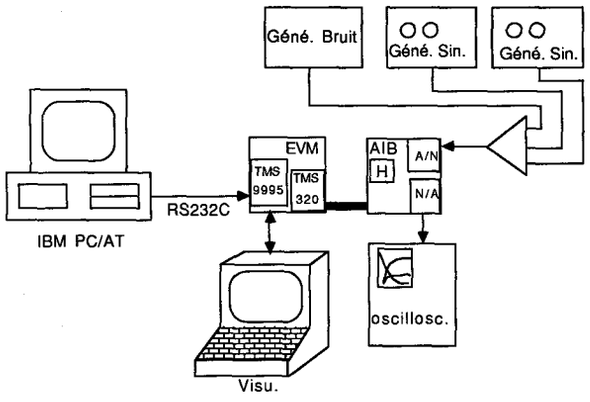


Fig. 5. – Schéma de la manipulation.

PROM qui gère le microprocesseur de traitement du signal TMS 32010.

– Une carte AIB d'interface constituée par les convertisseurs analogique-numérique (A/N) et numérique-analogique (N/A) avec leurs filtres associés et une horloge programmable (H).

– L'ordinateur personnel IBM-PC servant de système de développement grâce à ses logiciels spécialisés décrits par la suite.

La figure 6 représente un organigramme des étapes de la programmation de l'algorithme, effectuée sur l'IBM-PC. L'assembleur croisé et l'éditeur de lien sont des logiciels associés au TMS 32010 et implantables sur PC.

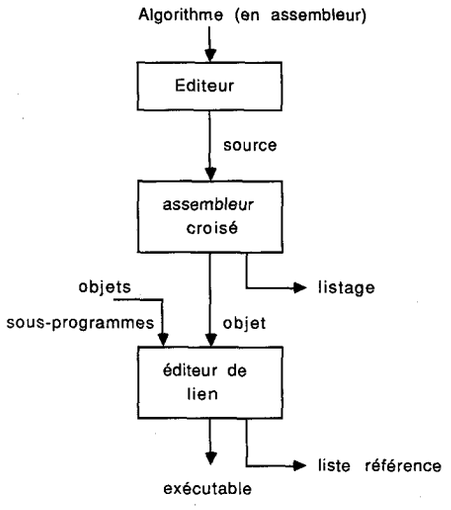


Fig. 6. – Chaîne de fabrication logicielle IBM-PC.

4.2. Méthodologie d'implantation

Le problème principal de l'implantation d'un algorithme sur un processeur travaillant en virgule fixe est l'estimation de la dynamique et la mise à l'échelle des variables. La méthodologie suivie est donnée dans le schéma de la figure 7.

La formulation mathématique de l'algorithme est d'abord traduite en langage FORTRAN sur une machine dont la dynamique et la précision sont limitées. En conséquence, la dynamique des variables de

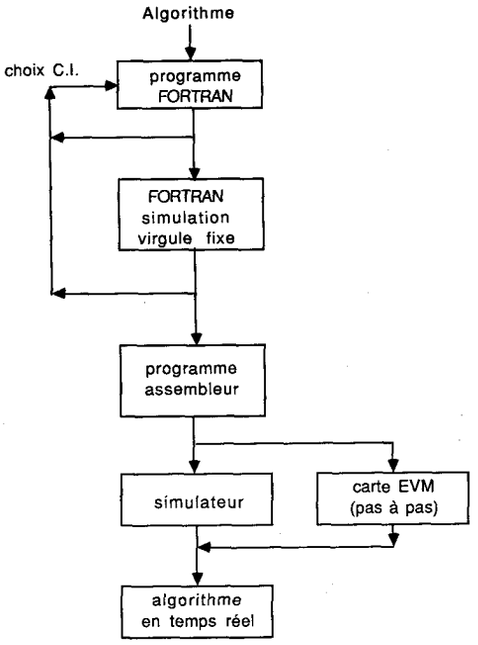


Fig. 7. – Chaîne de test de l'algorithme.

l'algorithme doit être estimée pour garantir le bon fonctionnement du programme. Cette dynamique dépend fondamentalement du signal à traiter et des conditions initiales.

Une étude théorique préalable permet d'évaluer l'excursion de certaines variables. Malheureusement, ce calcul fournit parfois des limites surestimées et, dans ce cas, il est nécessaire d'associer une procédure expérimentale pour déterminer la dynamique.

La deuxième étape consiste à refaire les simulations en virgule fixe, en vue de l'implantation sur TMS-320. Celui-ci fonctionne en arithmétique entière et avec une longueur de mot de 16 bits. Le choix des conditions initiales doit alors intervenir pour restreindre la dynamique des variables dans des limites raisonnables.

Nous n'avons pas reprogrammé l'algorithme FORTRAN avec des variables entières mais nous avons inséré un sous-programme de saturation et de troncature avec arrondi, ci-dessous mentionné :

```
C SATURATION ET TRONCATURE AVEC ARRONDI D'UNE VARIABLE
      SUBROUTINE SATRONC (VARIABLE,VMAX,NBITS)
C VMAX: VALEUR MAXIMUM EN 2**N
C NBITS:NOMBRE DE BITS APRES LA VIRGULE <16
            INTEGER*4 TVAR
C
C SATURATION
            IF(VARIABLE.GT.VMAX) VARIABLE=VMAX
            IF(VARIABLE.LT.-VMAX) VARIABLE=-VMAX
C TRONCATURE ET ARRONDI
            VARIABLE=VARIABLE*(2**NBITS)
            TVAR=IINT(VARIABLE)
            VARIABLE=FLOAT(TVAR)/2**NBITS
            RETURN
            END
```

Pour plus de commodité, les variables ont été normalisées par rapport au signal d'entrée, ce qui nous permet d'obtenir les valeurs maximales et, en consé-

quence, le nombre de bits nécessaires pour les parties entière et fractionnaire de la représentation en virgule fixe (VMAX et NBITS).

Il faut alors effectuer les simulations, vérifier les échelles et s'assurer que le comportement de l'algorithme, par rapport aux simulations en virgule flottante, reste acceptable. Ceci étant, une liste de résultats présentant l'évolution des variables est obtenue afin de servir de référence à l'étape la plus ardue, qui est la programmation en assembleur.

Le programme assembleur peut être exécuté pas à pas sur la carte EVM. Cependant, il est plus commode de faire appel au logiciel de simulation dans cette étape de mise au point.

Si les résultats obtenus sont conformes au fichier de référence de l'étape précédente, le programme est chargé sur la carte EVM et exécuté en temps réel.

Cette dernière étape permet de traiter des signaux réels et également d'effectuer un nombre d'itérations extrêmement important, ce qui permet d'étudier le comportement de l'algorithme à très long terme. Une réalisation en temps réel nous fournit donc un complément important pour l'analyse des performances et de la stabilité des algorithmes MCR.

5. Résultats

Nous avons mis en œuvre les filtres de prédiction d'ordre 2 et 4 et notre analyse est basée sur deux cas typiques :

– Le cas stationnaire avec une ou deux sinusoïdes à l'entrée, dont quelques résultats sont montrés dans la figure 8.

– Le cas où il y a un changement de fréquence ou une coupure du signal, illustré dans la figure 9.

L'algorithme démarre avec toutes les variables nulles sauf l'erreur cumulée. Le choix de cette condition initiale $E_a(0)$ doit respecter un compromis entre la vitesse de convergence et la dynamique limitée du gain. Une énergie d'erreur très petite augmente la vitesse de convergence mais peut provoquer l'instabilité si les gains atteignent des valeurs trop élevées.

Étant donné la relation (21), nous avons choisi la valeur de la constante de stabilisation de façon à ce que, après convergence, l'erreur cumulée reste au-dessus d'un seuil raisonnable.

Le facteur d'oubli w et le rappel à zéro des coefficients $(1-\delta)$ sont choisis de façon à garantir la stabilité à

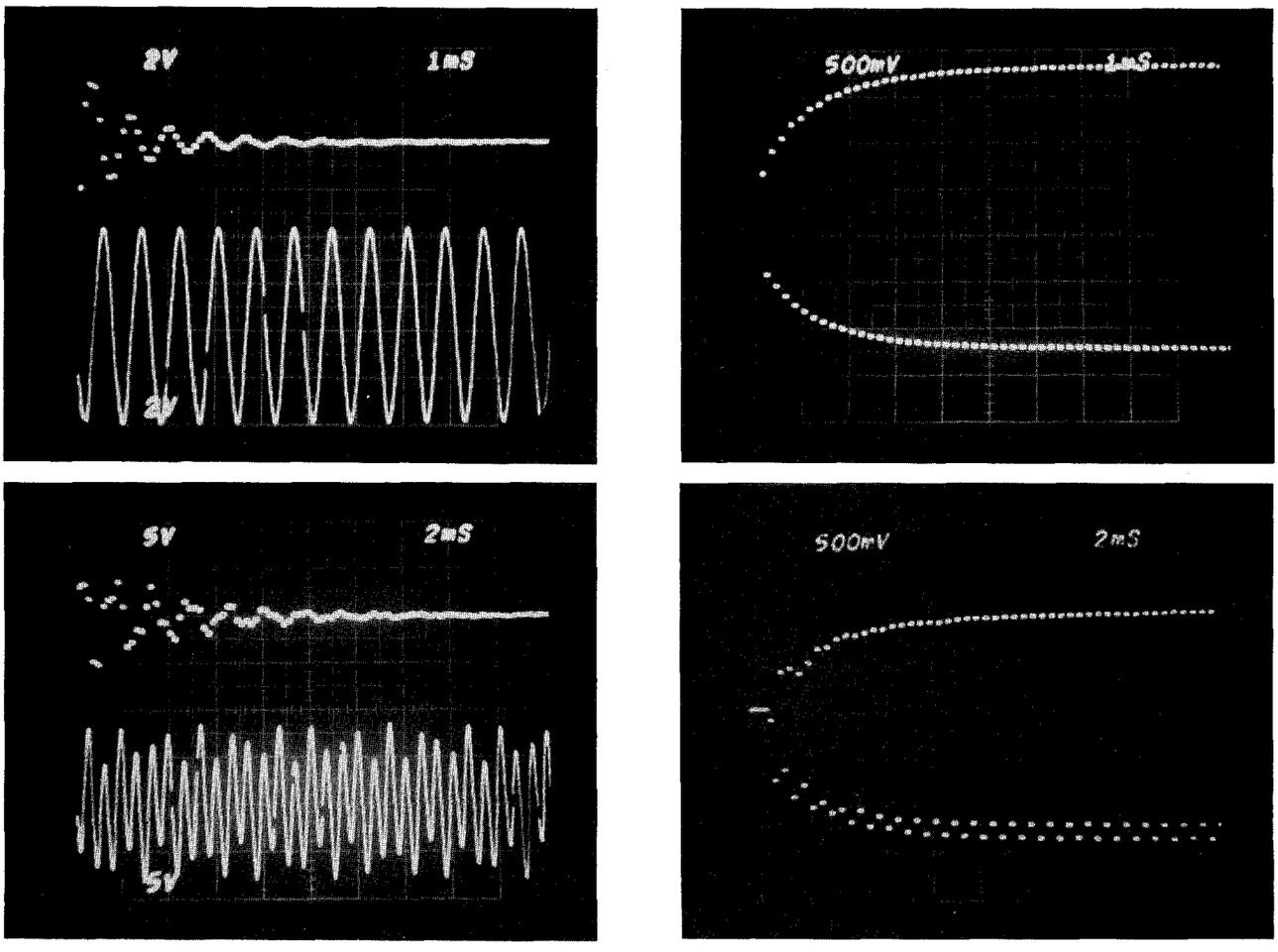


Fig. 8. – Résultats sur l'oscilloscope pour le cas stationnaire.

(A) N = 2, signal d'entrée (en bas) et erreur de prédiction (en haut).
(B) N = 2, évolution des coefficients.
(C) N = 4, signal d'entrée (en bas) et erreur de prédiction (en haut).
(D) N = 4, évolution des coefficients.

long terme, en prenant en compte l'expression (32) et les résultats de simulation, suivant la méthode représentée dans la figure 7. Les valeurs retenues sont alors :

$E_a(0) = 1/8$

$\mathscr{C} = 1/1024$

$w = 0{,}9375$

$(1-\delta) = 0{,}9990$

Il convient de noter que les performances de l'algorithme dépendent du soin apporté à la mise à l'échelle et du nombre de bits sur lesquels sont faits les calculs.

Nous avons utilisé des mots de 16 bits pour toutes les variables de l'algorithme. Les échelles et l'excursion maximale choisies pour les variables sont données dans le tableau ci-dessous :

Variable	Échelle	Excursion
Entrée $x(n)$	Q15	± 1
Erreurs de prédiction	Q14	± 2
Erreur cumulée	Q13	± 4
Coef. de prédiction	Q12	± 8
Gains	Q11	±16

CAS STATIONNAIRE

La figure 8 montre les courbes obtenues dans le cas où il n'y a pas de changement dans la sinusoïde d'entrée. Les deux premières correspondent au prédicteur d'ordre 2 et représentent le signal d'entrée de l'erreur de prédiction (*fig.* 8 A en bas et en haut) et l'évolution des coefficients de prédiction (*fig.* 8 B).

L'entrée est une sinusoïde de fréquence 1,25 kHz, échantillonnée à 10 kHz et d'amplitude 8 V crête-à-crête, étant donné que notre échelle ±1 correspond à ±10 V. La convergence est atteinte après environ 5 ms, équivalentes à 50 itérations de l'algorithme.

La constante de stabilisation empêche l'annulation de l'erreur cumulée mais produit, en contrepartie, une dégradation des performances pour les fréquences d'entrée proches de zéro et 5 kHz. La plage de fréquence où l'algorithme fonctionne normalement est donc inférieure à celle prévue par les équations (29) et (30), où l'effet de la constante de stabilisation n'est pas pris en compte. Nos échelles étant réglées pour un rapport signal/bruit (RSB), on note aussi une diminution de la plage de fréquence quand le bruit augmente, tel qu'il est montré dans le tableau III.

TABLEAU III

Plage de fréquence en fonction du rapport signal/bruit.

SNR (dB)	Marge de fréquence (Hz)
60	$330 < f < 4\,650$
50	$330 < f < 4\,650$
40	$360 < f < 4\,600$
30	$900 < f < 3\,900$
20	$1\,200 < f < 3\,700$

En présence d'un bruit élevé, il est nécessaire de modifier les échelles, en particulier pour les gains et l'erreur cumulée.

Le rappel à zéro sur les coefficients est indispensable pour la stabilité de l'algorithme. Nous avons, en effet, constaté des divergences après quelques minutes de fonctionnement quand $\delta = 0$, dues à l'accumulation des erreurs d'arrondi. Ce phénomène est encore plus important quand l'ordre du prédicteur est surdimensionné par rapport au nombre de sinusoïdes.

D'autre part, le biais sur les coefficients provoque évidemment une erreur résiduelle plus forte. Nous avons mesuré une erreur de l'ordre de 1 % quand $\delta = 0$ tandis que, pour $\delta = 10^{-3}$, elle varie entre 1,5 % pour $f = 2\,500$ Hz et 3 % pour les fréquences limites, avec SNR = 60 dB.

Nous pouvons aussi vérifier la convergence des deux coefficients dans la figure 8 B. L'accès à la mémoire de données du TMS nous permet d'obtenir les valeurs exactes des coefficients après convergence et nous avons constaté que l'erreur sur l'estimatuon de la fréquence est inférieure à 1 %.

Les figures 8 C et 8 D montrent des résultats analogues, obtenus pour un prédicteur d'ordre 4 avec deux sinusoïdes à l'entrée. Les fréquences utilisées sont 625 et 1 500 Hz, échantillonnées à 5 000 Hz.

Avec un rappel à zéro $(1-\delta) = 0{,}999$ l'algorithme a fonctionné en continu sans divergence, en présence de deux ou une seule sinusoïde à l'entrée. L'erreur résiduelle est de l'ordre de 3 % avec deux fréquences et de l'ordre de 1 % quand une des sinusoïdes est supprimée.

Dans l'évolution des quatre coefficients du prédicteur, on constate la superposition de deux courbes, représentant les coefficients du premier et du troisième degré (a_1, a_3), tandis que la valeur du dernier coefficient a_4 tend vers -1, ce qui correspond à $-1{,}25$ V sur l'oscilloscope avec l'échelle employée. Le filtre de prédiction a donc une fonction de transfert sous la forme d'un polynôme miroir et, en conséquence, ses zéros se situent sur le cercle unité.

CHANGEMENT DE SIGNAL

Dans la majorité des cas pratiques, l'algorithme adaptatif doit être robuste face à la suppression ou aux modifications du signal d'entrée. Nous avons effectué une première analyse de son comportement dynamique en temps réel en observant les cas de saut de fréquence et de coupure du signal.

La figure 9 A représente l'évolution des coefficients d'un prédicteur d'ordre 2 appliqué à une sinusoïde dont la fréquence est commutée de 1 680 à 1 140 Hz. La constante de stabilisation évite un débordement des gains au moment du changement et le facteur d'oubli utilisé, $w = 0{,}9375$, permet une convergence rapide.

Un exemple intéressant est celui où le signal d'entrée est supprimé pendant une certaine durée. Le rappel à zéro sur les coefficients permet d'obtenir des conditions initiales favorables à la reprise du signal.

Les deux dernières figures représentent l'évolution des coefficients du prédicteur d'ordre 2. Le signal d'entrée est une sinusoïde de 1 420 Hz, coupée périodiquement. L'adaptation est faite normalement en présence du rappel à zéro (*fig.* 9 B). Quand ce rappel est sup-

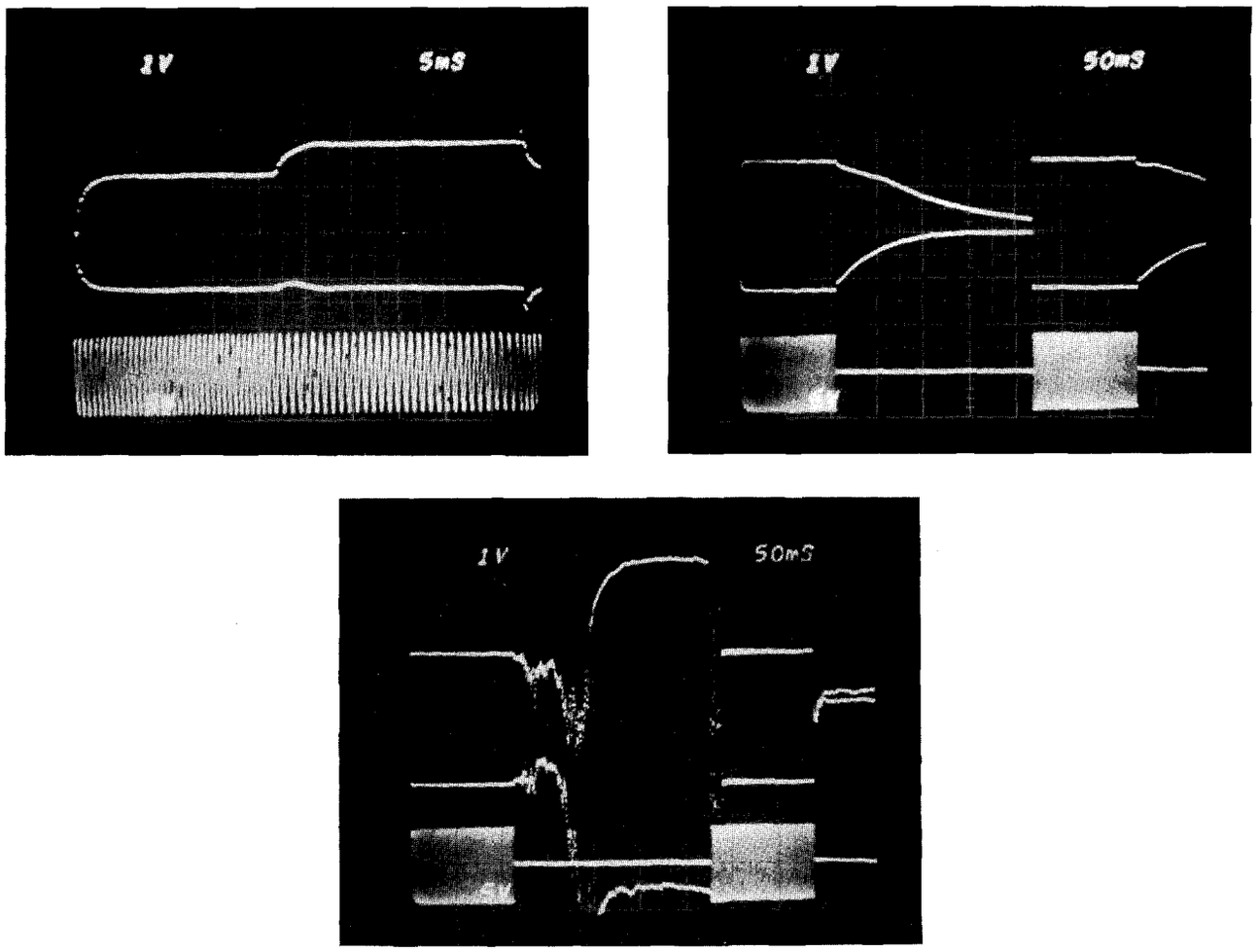


Fig. 9. — Signal d'entrée et évolution des coefficients avec changement de signal.
(A) Saut de fréquence.
(B) Coupure du signal avec rappel à zéro.
(C) Coupure du signal sans rappel à zéro.

primé, on observe la dérive des coefficients suivie de saturations et d'explosions au moment de la reprise du signal.

6. Conclusion

Nous avons présenté une implantation d'un algorithme des moindres carrés rapide en temps réel, appliqué à la prédiction linéaire des sinusoïdes bruitées.

Les principes de l'algorithme, en particulier les aspects concernant la stabilité, sont étudiés. Nous proposons l'utilisation d'une constante de stabilisation (20) et d'un rappel à zéro sur les coefficients (31) comme solution aux deux modes de divergence existants. L'expérience en temps réel confirme l'efficacité des deux méthodes dans les cas considérés.

Nous décrivons le montage expérimental et la méthodologie suivie, avec des indications sur les valeurs des paramètres et sur la mise en échelle des variables. Les résultats obtenus sont ensuite présentés et étudiés.

Une réalisation en temps réel démontre la possibilité d'avoir des algorithmes MCR stables en fonctionnement continu et ouvre des perspectives sur les applications pratiques de ces algorithmes.

Remerciements

Nous remercions vivement M. Maurice Bellanger pour sa collaboration tout au long de ce travail et ses conseils pour la rédaction de cet article.

Manuscrit reçu le 3 *septembre* 1987.
Version définitive le 25 *janvier* 1989.

BIBLIOGRAPHIE

[1] O. Macchi et M. Bellanger, Le point sur le filtrage adaptatif transverse, 11e *Colloque GRETSI*, Nice, juin 1987, p. 1G-14G.

[2] M. Bellanger, *Analyse des Signaux et Filtrage Numérique Adaptatif*, Masson, Paris, 1989.

[3] L. LJUNG, M. MORF et D. FALCONER, Fast calculation of gain matrices for recursive estimation schemes, *Int. Journal of Control*, 27, Janvier 1978, p. 1-19.

[4] G. CARAYANNIS, D. MANOLAKIS et N. KALOUPTSIDIS, A fast sequential algorithm for least-squares filtering and prediction, *IEEE Transactions of ASSP*, ASSP-31, n° 6, Dec. 1983, p. 1394-1402.

[5] R. ALCANTARA, Implantation d'algorithmes rapides sur des processeurs de traitement du signal, *Thèse de Docteur-Ingénieur*, ENST, Sept. 1986.

[6] J. L. BOTTO, Étude des algorithmes transversaux rapides : Application à l'annulation des échos acoustiques pour l'audioconférence, *Thèse de Docteur-Ingénieur*, Université de Rennes, mai 1986.

[7] R. ALCANTARA, J. PRADO, C. GUEGUEN, J. BOUDY et G. FAVIER, Simulation sur ordinateur des effets de quantification dans les algorithmes de filtrage adaptatifs, 11^{e} *Colloque GRETSI*, Nice, juin 1987, p. 772-775.