1/99

République Algérienne Démocratique et populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

ECOLE NATIONALE POLYTECHNIQUE
D.E.R DE GENIE ELECTRIQUE ET INFORMATIQUE
DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

Pour l'obtention du diplôme d'Ingénieur d'Etat en Electronique

Thème

Etude d'un annuleur d'écho acoustique

Proposé et dirigé par :

Mr. A. BELOUCHRANI
Mr. L. ABDELOUEL

Etudié par :

Mr. ABDULHAMID SAID
Mr. BELGACEM MONCEF

Promotion : JUIN 1999

E.N.P. 10, Avenue HASSEN BADI –EL-HARRACH– ALGER

République Algérienne Démocratique et populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

ECOLE NATIONALE POLYTECHNIQUE
D.E.R DE GENIE ELECTRIQUE ET INFORMATIQUE
DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

Pour l'obtention du diplôme d'Ingénieur d'Etat en Electronique

Thème

Etude d'un annuleur d'écho acoustique

Proposé et dirigé par :

Mr. A. BELOUCHRANI
Mr. L. ABDELOUEL

Etudié par :

Mr. ABDULHAMID SAID
Mr. BELGACEM MONCEF

Promotion : JUIN 1999

E.N.P. 10, Avenue HASSEN BADI –EL-HARRACH– ALGER

Remerciements

Nous remercions le seigneur de nous avoir donné la force et la volonté d'effectuer ce travail.

Nous tenons à exprimer nos plus sincères remerciements à Mr. BELOUCHRANI et à Mr. ABDELOUEL pour nous avoir offert la possibilité d'explorer ce domaine, pour leurs précieux conseils ainsi que leurs aides sans réserve tout au long de ce travail.

Enfin, nous remercions tous les professeurs du département ELECTRONIQUE, ainsi que tous ceux qui ont contribué de prés ou de loin à la réalisation de ce projet.

A. SAID – B. MONCEF

Dédicace

Je dédie ce travail à :

Mon père et à ma mère, qui étaient toujours présents à mes cotés et qui m'ont suivi et soutenu durant toutes mes études.

Ma sœur Rym ainsi qu'à Sabrina qui m'ont aidé durant toute l'année.

Je remercie spécialement mon ami d'enfance Larbi.

Je remercie également mes amis : Amar, Assia, Fiçal, Fouad, Hichem, Hichem, Khaled, Salima, Salim, Messaoud et Mounia.

A.Said

Dédicace

Je dédie ce travail à :

Ma mère, mon père, mes frères Yacine et Mehdi ainsi qu'à Assia, qui m'ont aidé et soutenu durant toute l'année.

Cette dédicace est adressée aussi à ma grand mère, mes oncles, mes tantes et à tous mes cousins et cousines.

Je tiens à remercier tout particulièrement Mr et Madame Kaidi ainsi que Mr et Madame Abdulhamid pour m'avoir soutenu tout le long de ce travail.

Je remercie également mes amis : Assira, Hichem, Larbi, Raffik, Ramzi, Messaoud, Salima, Mounia, Samir, Amar, Khaled, Fiçal, Hichem, ,Salim, Rym, Salim, Fouad.

B.Moncef

ملخص

في بعض التطبيقات كإلغاء الصد ى السمعي، من الضروري مماثلة و بطريقة متكيفة نظم ذات عدد هائل من العوامل المرشحة.إن مثل هذا العد دمن العوامل، يسبب في مثل هذه الحالة، تضخم حجم الحسابات و تأخيرا زمنيا غير معيقا.

إن إنشاء الخوارزميات المتكيفة في مجال التواترات(GMDFα) باستعمال ترشيح سـريع فـي نفس المجال ، و هيكل بالتجزئة يساعد على إزالة هذه العوائق.

Abstract

In many applications domains, as the acoustic echo cancellation, we have to identify long impulsional response systems by an adaptive way. A such number of coefficients provoque prohibitive arithmetic complexity and time delay.

The implementations of adaptif algorithms in the frequency domain (GMDFα), using fast convolution in the same domain and a multi-bloc structure , solve this problems .

Résumé

Dans de nombreux domaines d'applications, tels l'annulation d'écho acoustique, il est nécessaire d'identifier de façon adaptive des systèmes à réponse impulsionnelle longue. Un tel nombre de coefficients introduit une charge de calcul et un délai de traitement prohibitif.

L'implémentation d'algorithmes adaptatifs dans le domaine fréquentiel (GMDFα), utilisant la convolution rapide dans ce même domaine et une structure par bloc , permet de pallier ces problèmes .

Mots clefs

OLA : Over-lap add

OLS : Over-lap save

WOLA : Weighed over-lap add

GMDF : Generalized multi-delay adaptive filter

FFT : Fast Fourier Transform

Split-radix : Algorithme de FFT rapide

Filtrage fixe: Convolution par des coefficients fixes

Filtrage adaptative: Convolution par des coefficients qui s'adaptent

MCR : Moindres Carrés Récurssifs

LMS : Least Mean Square

BLMS : Bloc LMS

NLMS :Normalized LMS

FELMS : Fast Exact LMS

FTF : Fast Transversal Filter

Téléconférence :Communication entre deux salles distantes

Sommaire

Liste des Figures

Chapitre 1

Introduction

L'identification en temps réel de réponses impulsionnelles longues au moyen d'algorithmes de filtrage adaptatif est un problème très général que l'on rencontre dans de nombreuses applications en traitement du signal et en commande des systèmes. Une application typique dans le domaine des télécommunications où ce problème est posé est l'annulation de l'écho acoustique en téléphonie main libre dont un exemple typique est la téléconférence.

Considérons la situation suivante (figure 1.1), où deux salles distantes sont reliées à travers un couple de haut – parleurs et de microphones, le signal provenant du haut – parleur filtré par le canal acoustique de la salle (géométrie de la salle, disposition des équipement dans la salle, déplacement aléatoires des gens dans la salle) est ré – injecté dans le microphone. De ce fait, le locuteur éloigné va recevoir un écho de sa voix d'autant plus gênant qu'il se manifeste avec un retard perceptible. Afin de réduire ce couplage, l'annuleur d'écho soustrait du signal transmis au microphone une estimée du l'écho issue du haut

– parleur. La réponse de la salle variant dans le temps, un filtrage adaptatif s'impose. Le signal d'entrée *x(n)* (haut – parleur) est filtré par une estimée de la réponse impulsionnelle associée au chemin de l'écho minimisant l'énergie du signal d'erreur *e(n)* (différence entre l'écho estimé *y'(n)* et le signal transmis au microphone *d(n)*). Nous traitons dans ce rapport, un problème d'estimation linéaire dont la solution passe par la minimisation d'une erreur quadratique moyenne. La résolution de ce problème est ardue car :

- la réponse impulsionnelle modélisant le canal d'écho est très longue, de quelques centaines de coefficients (téléphonie main libre) à quelques milliers de coefficients (téléconférence). De plus, la vitesse de convergence devient cruciale.

- les mouvement des personnes et d'objets dans la salle, provoquent des variations de la réponse impulsionnelle rapides difficilement prédictibles (non stationnarité du système). En conséquences l'algorithme doit réagir de façon rapide à ces variations. Ceci est d'autant plus contraignant que la réponse est longue.

- les signaux d'entrée ne sont pas stationnaires à l'échelle de la réponse impulsionnelle et présentent une importante dynamique spectrale .

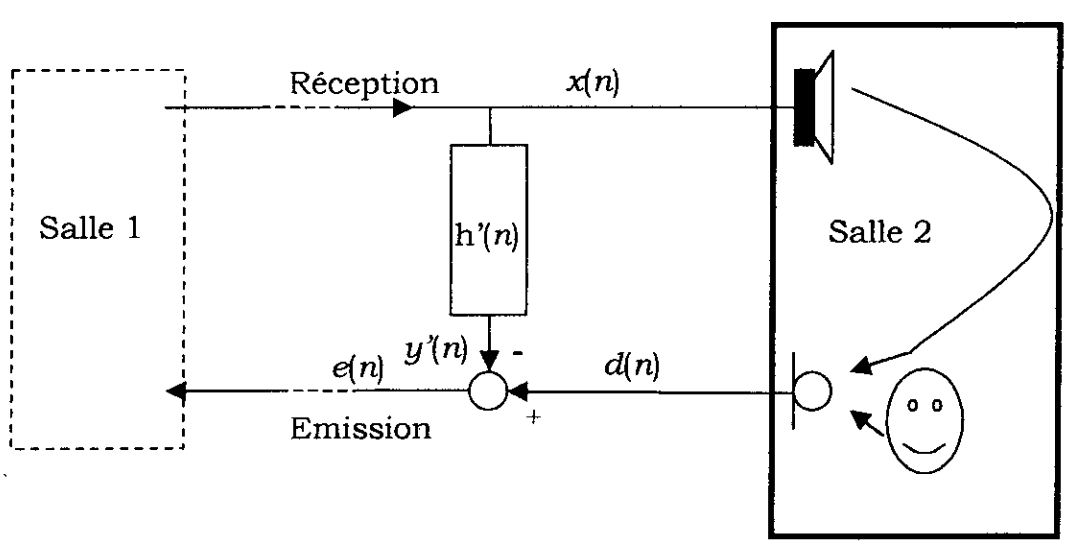

Figure 1.1 : Procédure d'annulation d'écho acoustique

- La contrainte du délai qui doit être maintenue le plus faible possible afin de ne pas perturber la communication.

Cette application a donné naissance à plusieurs recherches qui ont abouti au développement de nouveaux algorithmes de filtrage adaptatif transverse supérieur à l'algorithme du gradient stochastique [GIL 92], classiquement utilisé dans cette application en raison de sa faible complexité de calcul bien que sa convergence soit lente.

Les algorithmes de type moindres carrés transversaux rapides (MCR) convergent beaucoup plus rapidement que le gradient stochastique, mais leur complexité est sensiblement plus élevée. Les algorithmes de Newton transversaux rapides (FNTF), apparus dans le début des années 1990, proposent une complexité réduite par rapport aux MCR avec des performances pratiquement équivalentes pour la parole.

Une autre approche suivie est la recherche de structures de calculs par blocs d'échantillons moins complexes que le filtrage adaptatif transverse classique. Le filtre étant figé sur la durée du bloc, des techniques de convolution rapide permettent de réduire significativement la complexité. De plus la convergence peut être nettement accélérée par l'utilisation de la densité spectrale du signal estimée sur le bloc.

Dans ce contexte, deux famille d'algorithme ont été développées :

- Algorithmes à grande vitesse de convergence.
- Algorithmes à faible complexité.

1.1 Algorithmes à grande vitesse de convergence

1.1.1 La famille des algorithmes de Newton rapides

La résolution directe des équations des Moindres Carrés Récursifs (MCR) implique l'inversion de l'estimée de la matrice d'autocorrélation des entrées. L'utilisation du lemme d'inversion matricielle [HAY 91] permet une mise à jour direct de l'inverse de la matrice d'autocorrélation, au moyen d'une quantité appelée gain de Kalman, ce qui réduit la charge de calcul.

Pour une classe assez générale de filtres adaptatifs (RIF) incluant les algorithmes GS (Gradient Stochastique) et MCR, les équations de filtrage et de mise à jour s'écrivent :

Filtrage : $e(n)=d(n)-\mathrm{h}(n-1)^{\mathrm{T}}\,\mathrm{x}(n)$ (1.1)

Mise à jour : $\mathrm{h}(n)=\mathrm{h}(n-1)-\mathrm{C}_N(n)e(n)$ (1.2)

où $\mathrm{h}^{\mathrm{T}}(n)=[h_1(n),\ldots,h_N(n)]$ et $\mathrm{x}(n)=[x(n),\ldots,x(n-N+1)]^{\mathrm{T}}$.

L'expression théorique du vecteur de gain d'adaptation $\mathrm{C}_N(n)$ dépend de l'algorithmes utilisé. Considérons d'abord l'algorithme GS, sous forme normalisée (nommée NLMS), qui est utilisé en pratique ; le vecteur de gain s'écrit :

$$\mathrm{C}_N(n)=-(\mu/\mathrm{P}_x(n))\,\mathrm{x}(n) \qquad (1.3)$$

où $\mathrm{P}_x(n)$ est une estimation courante de la puissance du signal d'entrée et μ le pas d'adaptation.

Considérons maintenant l'algorithme MCR, l'expression théorique du vecteur de gain d'adaptation (gain de Kalman) s'écrit :

$$\mathrm{C}_N(n)=-\mathrm{R}^{-1}{}_{\mathrm{N}}(n)\,\mathrm{x}(n) \qquad (1.4)$$

où $R_N(n)$ est l'estimation courante de la matrice de covariance (de taille $N*N$) du signal d'entrée :

$$R_N(n)=\lambda R_N(n-1)+ x(n)x(n)^T \qquad (1.5)$$

$\lambda<1$ est le facteur d'oubli de l'algorithme. Nous rappelons que la forme transverse rapide de l'algorithme MCR effectue de façon efficace la mise à jour du vecteur de gain (1.4) avec une complexité de l'ordre de ($6N$) multiplications au minimum.

L'idée de base de la nouvelle famille des algorithmes de Newton rapides est que le signal d'entrée peut être modélisé comme un processus AR d'ordre L inférieur à N, alors nous pouvons construire une approximation optimale de la matrice de covariance $R_N(n)$ d'ordre N par un mécanisme d'extrapolation simple à partir des L premières valeurs de la covariance estimée de l'entrée. Il en résulte que la mise à jour du vecteur de gain de Kalman (1.4) de taille N peut être obtenue à partir des prédicteurs d'ordre L correspondant à un algorithme MCR du même ordre. Pour les formes transverses rapides des MCR, ceci conduit à une importante réduction du volume des calculs lorsque L est très inférieur à N.

1.1.2 Technique de blanchissement

De par leur simplicité, leur faible complexité ainsi que leur robustesse, les algorithmes de descente de gradient GS sont des outils privilégiés pour l'implantation en temps réel. Cependant, la vitesse de convergence de ces algorithmes est inversement proportionnelle à la longueur du filtre adaptatif et à la dispersion des valeurs propres de la matrice d'autocorrélation du signal d'entrée du système. L'insertion d'un filtre blanchisseur dans le schéma d'identification, permet de décorréler le signal d'entrée et ainsi améliorer et de façon significative la vitesse de convergence.

Le blanchiment est introduit dans cette technique sans pour autant modifier la relation entrée-sortie, en séparant les opérations de filtrage et d'adaptation. le signal d'entrée est directement appliqué à l'entrée de l'annuleur

(figure 1.2) et le signal blanchi n'intervient que là où il est nécessaire (partie adaptation).

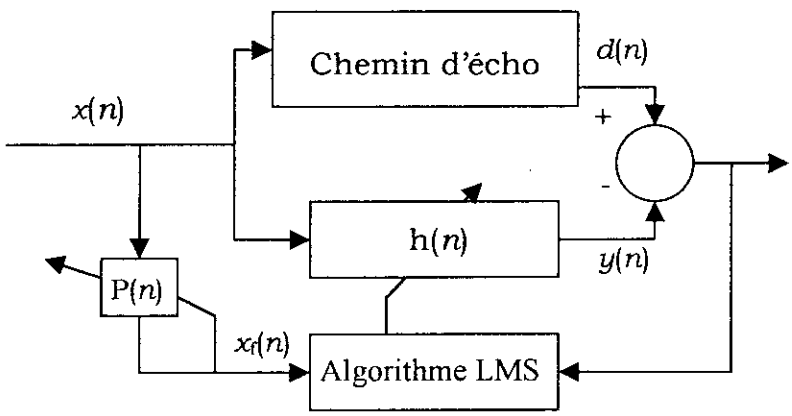

Figure 1.2 : Technique de blanchiment

1.2 Algorithmes de faible complexité

plusieurs familles d'algorithmes par blocs ont été développées avec comme principal objectif la réduction de complexité. Citons pour mémoire le filtrage adaptatif en sous-bandes, où les signaux sont traités séparément dans des bandes de fréquences adjacentes à une cadence inférieure à la fréquence d'échantillonnage originale. Des formes par blocs de l'algorithme GS proposées initialement (par exemple le bloc-LMS), ont évolué vers des formes d'algorithmes dites à petits blocs adaptées au contexte d'annulation d'écho acoustique.

1.2.1 Filtrage adaptatif GS par blocs « *Blocs-LMS* »

Le bloc-LMS est construit à partir du GS en imposant que les coefficients du filtre adaptatif restent fixes sur chaque bloc de taille *N*. On définit la matrice d'échantillons de taille *N*N* :

$$\mathbf{x}(n)=[x(n-N+1),\ x(n-N+2),\ldots,\ x(n)] \qquad (1.6)$$

Le filtre h étant maintenu fixe entre les instants $(n\text{-}N+1)$ et (n), le vecteur des N erreurs successives (différentes de celles du GS) $E(n)=[e(n\text{-}N+1), \ldots, e(n)]^T$ satisfait à :

$$E(n)=D(n)-x^T(n)h(n\text{-}N) \tag{1.7}$$

avec $D(n)=[d(n\text{-}N+1), d(n\text{-}N+2),\ldots, d(n)]^T$. Imposant le pas d'adaptation μ fixe sur tout le bloc, l'équation d'adaptation par bloc du filtre s'écrit :

$$h(n)=h(n\text{-}N)+ (\mu/N)\, x^T(n)E(n) \tag{1.8}$$

Cet algorithme permet une importante réduction de complexité par rapport au GS standard (que l'on trouve pour $N=1$) par l'emploi de techniques de convolution rapide, mais sa vitesse de convergence est dégradée par rapport à celle du GS sur des signaux d'entrée corrélés. De nouvelles formes d'algorithmes évitent cet inconvénient.

1.2. 2 Algorithme GS exact par blocs (FELMS) « *Fast exact LMS* »

Cet algorithme est construit en itérant sur le temps les équations du GS standard. Avec des notations analogues à celles définies précédemment, on obtient les équations suivantes :

$$E(n)=G(n)[D(n)-x^T(n)h(n\text{-}N)] \tag{1.9}$$

$$h(n)=h(n\text{-}N)+ \mu\, x^T(n)E(n) \tag{1.1O}$$

avec

$$G(n)=[\mu S(n)+ I_N]^{-1}$$

où $S(n)$ est une matrice triangulaire inférieure avec $S_i(n)=x^T(n)x(n\text{-}i)$, $i = 0,1,\ldots,(n\text{-}1)$ contenant des termes de corrélation de l'entrée.

La matrice G(n) est donc facilement calculable et si on la remplace par la matrice identité I_N on retrouve l'algorithme Bloc-LMS, qui peut être considéré comme une approximation du FELMS.

L'algorithme FELMS est par construction équivalent au GS standard : la suite des erreurs de filtrage produite et le comportement adaptatif (convergence, capacité de poursuite) sont identiques. Comme dans le cas du Bloc-LMS, la réduction de complexité peut être obtenue par l'intermédiaire de techniques de convolution rapide.

1.2.3 Algorithmes de filtrage fréquentiel à faible retard

Dans des formulations plus anciennes que celles présentées dans les paragraphes précédents, le problème inhérent au traitement par blocs était résolu par segmentation de la réponse impulsionnelle du filtre adaptatif. Dans cet algorithme (objet de notre rapport) la réduction de charge de calcul est obtenue au moyen de la technique dite Over Lap-Save (OLS) de convolution rapide dans le domaine fréquentiel. On dispose alors d'un outil qui allie à une faible complexité une souplesse dans le choix du délai de traitement. On montrera dans cette étude que cette structure est un membre particulier d'une famille plus générale, qui par extension de l'appellation sera dénommée Generalized Multi-Delay Adaptive Filter (GMDF).

Dans un premier temps, nous étudierons des méthodes de convolution rapide dans le domaine fréquentiel, qui serviront à l'élaboration du filtre adaptatif qui fait l'objet de notre rapport.

En deuxième lieu, nous ferons l'étude et l'implémentation rapide des équations du filtre adaptatif GMDFα.

Enfin, nous finirons ce rapport par des simulations qui porteront sur l'efficacité du filtre présenté en annulation d'écho acoustique.

Chapitre 2

Algorithmes de calcul de convolution rapide dans le domaine fréquentiel

Le filtrage transverse (RIF) est l'une des opérations les plus rencontrées en traitement du signal. L'évaluation directe d'un filtre implique le calcul de convolutions discrètes de longueurs égales à la taille du filtre. Pour des filtres long, ceci se traduit par une complexité arithmétique élevée et un délai de traitement prohibitif. Ceci a conduit à l'utilisation de moyens plus efficaces et rapides tel la segmentation du filtre à évaluer et l'exploitation des propriétés de techniques de convolution rapide.

Dans ce chapitre, nous exposerons les techniques fondées sur la convolution circulaire, dont découle la famille du filtre adaptatif étudié par la suite. Une première partie contiendra une description détaillée des différents algorithmes de convolution circulaire classiques, où le filtre est traité dans sa globalité et qui seront dénommés « algorithmes mono – bloc ». Par la suite, la formulation sera étendue au cas où le filtre est segmenté en petits blocs « algorithmes multi – blocs ».

Dans la première partie de ce chapitre, nous introduisons quelques propriétés des matrices circulantes utiles pour les développements qui suivent.

Les variables fréquentielles seront représentées par des lettres majuscules et les variables temporelle en lettres minuscules. Les matrices est les vecteurs seront représentées par des caractères gras.

2.1 Matrices circulantes

Une matrice Ψ de dimension $2N$ est dite matrice circulante si et seulement si : $\psi_{ij} = \psi_{<i-j>2N}$, où $<x>2N$ est le résidu de x modulo $2N$. Les colonnes ainsi que les lignes d'une telle matrice se déduisent les unes des autres par circulation tel que le dernier élément de la colonne n devient le premier élément de la colonne $n+1$.

$$\Psi = \begin{pmatrix} \psi_0 & \psi_{2N-1} & \cdots & \psi_2 & \psi_1 \\ \psi_1 & \psi_0 & \cdots & \psi_3 & \psi_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \psi_{2N-2} & \psi_{2N-3} & \cdots & \psi_0 & \psi_{2N-1} \\ \psi_{2N-1} & \psi_{2N-2} & \cdots & \psi_1 & \psi_0 \end{pmatrix} \qquad (2.1)$$

De plus, l'une des propriétés principales des matrices circulantes est qu'elles peuvent s'écrire sous la forme :

$$\Psi = \mathbf{W}^{-1}\,\mathbf{X}\,\mathbf{W} \qquad (2.2)$$

où $\mathbf{X}$ représente la matrice diagonale dont les éléments sont les coefficients de la TFD de la première colonne de Ψ et $\mathbf{W}$, la matrice transformation de Fourier discrète (TFD) de dimension $2N$ est tel que :

$$\mathbf{W}_{kl} = \exp(-j2\pi\, kl/2N) = \exp(-j\,\Omega_k\, l)$$

De cette décomposition découle ces deux propriétés :

1- Les matrices circulantes commutent.

2- Le produit de deux matrices circulantes est une matrice circulante.

2.2 Convolution rapide dans le domaine fréquentiel

Tout algorithme de filtrage rapide doit réaliser une convolution linéaire entre une réponse impulsionnelle $\{h_n\}$ de longueur finie et une séquence d'entrées $\{x_n\}$ de longueur infinie.

De façon plus formelle, on considère la sortie y_n du filtre à l'instant n. Cette sortie découle de la convolution discrète de la séquence de réponse impulsionnelle $\{h_n\} = [h_0,\ldots,h_{N-1}]$ par la séquence des entrées $[x_n,\ldots,x_{n-N+1}]$ donc :

$$y_n = \sum_{k=0}^{N-1} h_k \, x_{n-k} = \mathrm{h}^{\mathrm{T}} \mathrm{x}_{\mathrm{n}} \qquad (2.3)$$

tel que : $\mathbf{h} = [h_0,\ldots,h_{N-1}]^T$

$\mathbf{x}_n = [x_n,\ldots,x_{n-N+1}]^T$

où T représente le transposé.

Sur un bloc de N sorties évaluées sur l'intervalle $[n+1, n+N]$ on peut écrire :

$$\begin{pmatrix} y_{n+1} \\ y_{n+2} \\ \cdot \\ \cdot \\ \cdot \\ y_{n+N-1} \\ y_{n+N} \end{pmatrix} = \begin{pmatrix} x_{n+1} & x_n & \cdots & x_{n-N+3} & x_{n-N+2} \\ x_{n+2} & x_{n+1} & \cdots & x_{n-N+4} & x_{n-N+3} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{n+N-1} & x_{n+N-2} & \cdots & x_{n+1} & x_n \\ x_{n+N} & x_{n+N-1} & \cdots & x_{n+2} & x_{n+1} \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ \cdot \\ \cdot \\ \cdot \\ h_{N-2} \\ h_{N-1} \end{pmatrix} \qquad (2.4)$$

Cette formulation matricielle va être utilisée dans ce qui suit pour en déduire les algorithmes de convolution rapide dans le domaine fréquentiel exploitant les propriétés des matrices circulantes.

2.2.1 Convolution rapide par OLS *«Over Lap-Save»*

Partant de la formulation matricielle définit précédemment, on peut obtenir une matrice circulante en faisant une extension judicieuse de la première colonne des entrées dans le passé, de la façon suivante :

$$\begin{pmatrix} y_{n-N+1} \\ y_{n-N+2} \\ \cdot \\ y_{n-1} \\ y_n \\ y_{n+1} \\ \cdot \\ \cdot \\ y_{n+N} \end{pmatrix} = \begin{pmatrix} x_{n-N+1} & x_{n+N} & \cdot & \cdot & x_{n+2} & x_{n+1} & x_n & \cdot & \cdot & x_{n-N+2} \\ x_{n-N+2} & x_{n-N+1} & \cdot & \cdot & x_{n+3} & x_{n+2} & x_{n+1} & \cdot & \cdot & x_{n-N+3} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{n-1} & x_{n-2} & \cdot & \cdot & x_{n+N} & x_{n+N-1} & x_{n+N-2} & \cdot & \cdot & x_n \\ x_n & x_{n-1} & \cdot & \cdot & x_{n-N+1} & x_{n+N} & x_{n+N-1} & \cdot & \cdot & x_{n+1} \\ x_{n+1} & x_n & \cdot & \cdot & x_{n-N+2} & x_{n-N+1} & x_{n+N} & \cdot & \cdot & x_{n+2} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{n+N} & x_{n+N-1} & \cdot & \cdot & x_{n+1} & x_n & x_{n-1} & \cdot & \cdot & x_{n-N+1} \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ \cdot \\ h_{N-1} \\ 0 \\ 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix} \tag{2.5}$$

On remarque dans cette formulation, que la réponse impulsionnelle de longueur N, a été étendue de $2N$ éléments par N zéros. De cette relation on peut écrire :

$$\boldsymbol{y}_n = \Psi_n \, \mathbf{h'} \tag{2.6}$$

où $\mathbf{h'}=[\mathbf{h}_N \ , \mathbf{0}_N]^T$, où $\mathbf{0}_N$ est le vecteur nul de longueur N.

D'après l'équation (2.5), on remarque que les $N-1$ premiers termes de sortie $[y_{n-N+1},\ldots,y_{n-1}]$ résultent d'une convolution circulaire, tant dis que les $N+1$ suivants $[y_n,\ldots,y_{n+N}]$ résultent d'une convolution linéaire du bloc d'entrées $[x_{n-N+1},\ldots,x_{n+N}]$ par le filtre $\mathbf{h'}$.

La figure (2.1) schématise la relation (2.5) et montre clairement le recouvrement des données pour former le bloc de $2N$ entrées qui permettent d'obtenir la convolution linéaire.

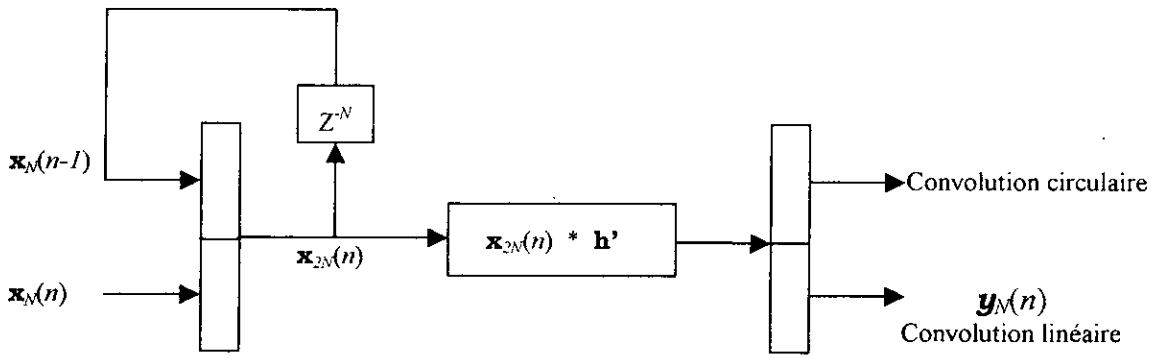

Figure 2.1 : Recouvrement des entrées

L'intérêt majeur de cette démarche est l'utilisation des propriétés de décomposition inhérente aux matrices circulantes dans le domaine fréquentiel. En effet, la matrice Ψ_n, circulante de dimension $2N$, est diagonalisable dans la base de Fourier :

$$\Psi_n = \mathbf{W}^{-1}\,\mathbf{X}_n\mathbf{W} \tag{2.7}$$

La matrice $\mathbf{X}_n$, est une matrice diagonale dont les éléments sont égaux à la transformée de Fourier de la première colonne de la matrice Ψ_n évaluées aux fréquences discrètes normalisées($\Omega_k = 2\pi k/2N$) où :

$$X_n(k) = \sum_{l=0}^{2N-1} x_{n+l-N+1} \exp(-j\Omega_k l) \tag{2.8}$$

L'organigramme de la technique OLS pourra s'écrira donc comme suit (figure 2.3) :

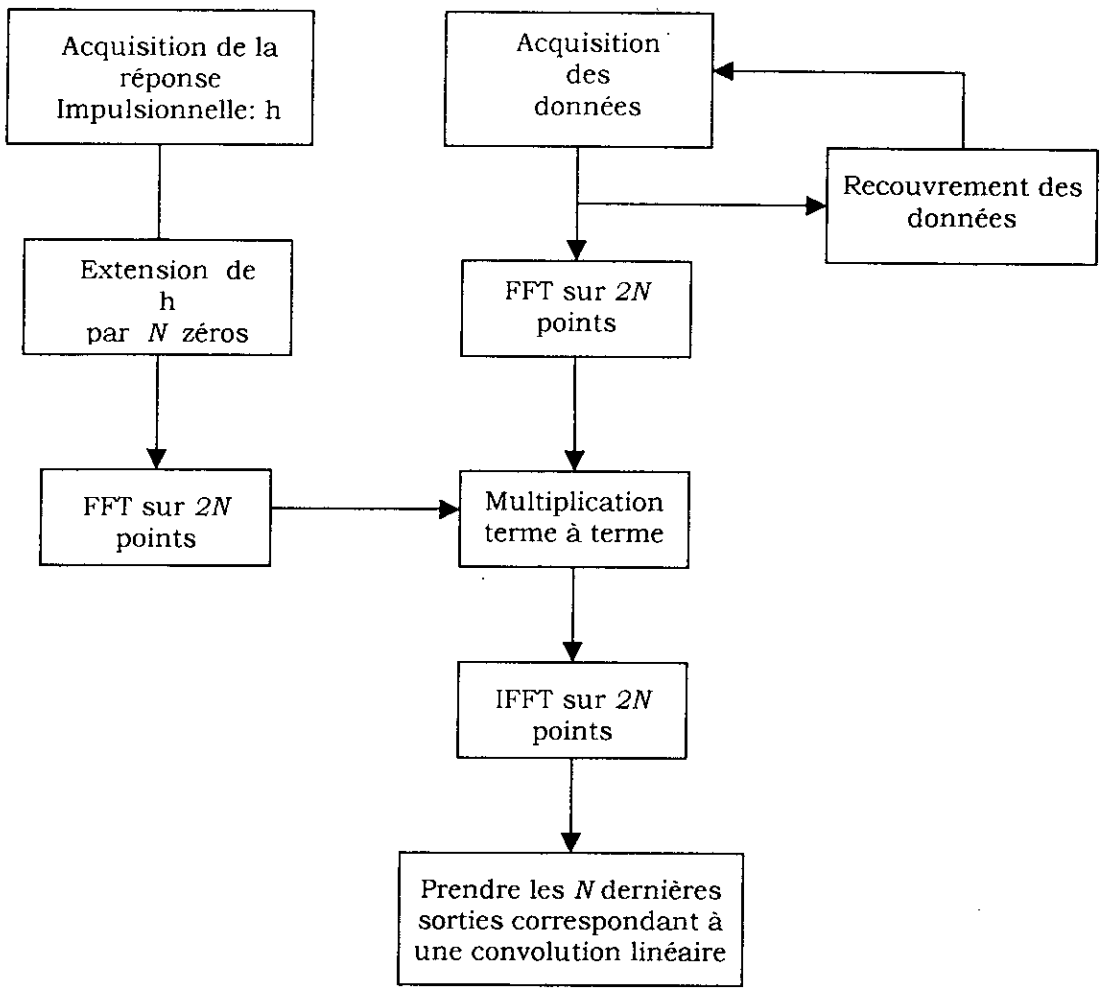

Figure 2.3 : Organigramme de la technique OLS

L'implémentation de cet algorithme en MATLAB nous a clairement montré que les résultats obtenus grâce à cet algorithme concordent exactement avec les résultats obtenus à l'aide de la convolution linéaire classique avec une réduction intéressante de charge de calcul .

Dans le cas de la convolution discrète le nombre de multiplications est de $2N$ et celui des additions de $N(N\text{-}1)$,tandis que pour la convolution OLS, exploitant déjà la réduction de charge de calcul en utilisant la FFT, on peut améliorer le gain de façon plus notable en utilisant l'algorithme FFT Split-Radix [DUH 86] de façon à avoir $[2N\log_2 N + 4]$ multiplications réelles et $[2N\log_2 N - 2N + 8]$ additions réelles avec $2N = 2^n$ (suite à une FFT de longueur $2N$, une FFT inverse de même longueur et de $2N$ multiplications complexes terme à terme).

Il est clair que le gain en opérations est largement significatif à mesure que la taille de la réponse impulsionnelle augmente (figure 2.4)

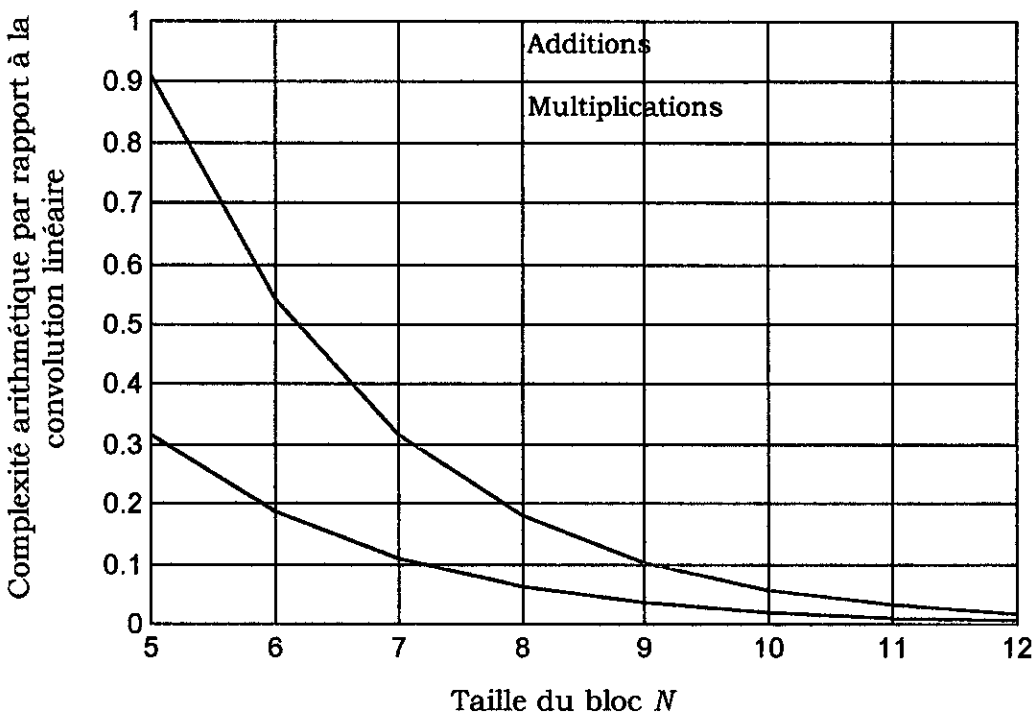

Figure 2.4 : Complexité par rapport à l'évaluation directe de la convolution

2.2.2 Convolution rapide par OLA *« Over Lap-Add »*

A partir de la relation matricielle (2.4), une deuxième approche de convolution rapide dans le domaine fréquentiel peut être exploitée en faisant une extension de N zéros dans la première colonne du bloc courant (les N premiers termes) puis un recouvrement des sorties par addition pour obtenir les termes correspondant à une convolution linéaire.

On obtient alors :

$$\Psi = \begin{pmatrix} 0 & x_{(s+1)N} & . & . & x_{sN+2} & x_{sN+1} & 0 & . & . & 0 \\ 0 & 0 & . & . & x_{sN+3} & x_{sN+2} & x_{sN+1} & . & . & 0 \\ . & . & . & . & . & . & . & . & . & . \\ 0 & 0 & . & . & x_{(s+1)N} & x_{(s+1)N-1} & x_{(s+1)N-2} & . & . & 0 \\ 0 & 0 & . & . & 0 & x_{(s+1)N} & x_{(s+1)N-1} & . & . & x_{sN+1} \\ x_{sN+1} & 0 & . & . & 0 & 0 & x_{(s+1)N} & . & . & x_{sN+2} \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ x_{(s+1)N} & x_{(s+1)N-1} & . & & x_{sN+1} & 0 & 0 & . & . & 0 \end{pmatrix} \quad (2.11)$$

Donc : $\boldsymbol{y'}_{sN} = \Psi_{sN}\,\mathbf{h'}$;

Le développement de cette formule montre clairement que le vecteur résultant $\boldsymbol{y'}_{sN}$ ne correspond qu'à une convolution partielle. Cependant, un examen attentif de ce développement montre que les termes manquants dans les N dernières composantes de $\Psi_{(s+1)N}\mathbf{h'}$, correspondent exactement aux N premières composantes de $\Psi_{sN}\mathbf{h'}$ comme le montre l'exemple suivant :

Soit un signal d'entrée composé de 4 échantillons (N=4) évalués au temps n $n = sN = 4s$, tel que $\mathbf{x} = [x_{4s+1}, x_{4s+2}, x_{4s+3}, x_{4(s+1)}]^T$.

D'après la relation $\boldsymbol{y'}_{sN} = \Psi_{sN}\,\mathbf{h'}$, on peut écrire :

$$\begin{pmatrix} y'_{4(s-1)+1} \\ y'_{4(s-1)+2} \\ y'_{4(s-1)+3} \\ y'_{4s} \\ y'_{4s+1} \\ y'_{4s+2} \\ y'_{4s+3} \\ y'_{4(s+1)} \end{pmatrix} = \begin{pmatrix} 0 & x_{4(s+1)} & x_{4s+3} & x_{4s+2} & x_{4s+1} & 0 & 0 & 0 \\ 0 & 0 & x_{4(s+1)} & x_{4s+3} & x_{4s+2} & x_{4s+1} & 0 & 0 \\ 0 & 0 & 0 & x_{4(s+1)} & x_{4s+3} & x_{4s+2} & x_{4s+1} & 0 \\ 0 & 0 & 0 & 0 & x_{4(s+1)} & x_{4s+3} & x_{4s+2} & x_{sN+1} \\ x_{4s+1} & 0 & 0 & 0 & 0 & x_{4(s+1)} & x_{4s+3} & x_{sN+2} \\ x_{4s+2} & x_{4s+1} & 0 & 0 & 0 & 0 & x_{4(s+1)} & x_{4s+3} \\ x_{4s+3} & x_{4s+2} & x_{4s+1} & 0 & 0 & 0 & 0 & x_{4(s+1)} \\ x_{4(s+1)} & x_{4s+3} & x_{4s+2} & x_{4s+1} & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Donc :

$$\begin{cases} y'_{4(s-1)+1} = h_1 x_{4(s+1)} + h_2 x_{4s+3} + h_3 x_{4s+2} \\ y'_{4(s-1)+2} = h_2 x_{4(s+1)} + h_3 x_{4s+3} \\ y'_{4(s-1)+3} = h_3 x_{4(s+1)} \\ y'_{4s} = 0 \\ y'_{4s+1} = h_0 x_{4s+1} \\ y'_{4s+2} = h_0 x_{4s+2} + h_1 x_{4s+1} \\ y'_{4s+3} = h_0 x_{4s+3} + h_1 x_{4s+2} + h_2 x_{4s+1} \\ y'_{4(s+1)} = h_0 x_{4(s+1)} + h_1 x_{4s+3} + h_2 x_{4s+2} + h_3 x_{4s+1} \end{cases}$$

Si on fait la même évaluation au temps $n = (s-1)N = 4(s-1)$, on obtient :

$$\begin{cases} y'_{4(s-2)+1} = h_1 x_{4s} + h_2 x_{4s-1} + h_3 x_{4s-2} \\ y'_{4(s-2)+2} = h_2 x_{4s} + h_3 x_{4s-1} \\ y'_{4(s-2)+3} = h_3 x_{4s} \\ y'_{4(s-1)} = 0 \\ y'_{4(s-1)+1} = h_0 x_{4s} \\ y'_{4(s-1)+2} = h_0 x_{4s-2} + h_1 x_{4s-3} \\ y'_{4(s-1)+3} = h_0 x_{4s-1} + h_1 x_{4s-2} + h_2 x_{4s-3} \\ y'_{4s} = h_0 x_{4s} + h_1 x_{4s-1} + h_2 x_{4s-2} + h_3 x_{4s-3} \end{cases}$$

On remarque que les termes manquant aux N dernières sorties (au temps d'évaluation $n=4s$) pour former une convolution linéaire sont les N premiers termes de l'évaluation précédente (pour $n=4(s-1)$).

Donc, en effectuant un recouvrement des sorties par addition, on trouve :

$$
\begin{cases}
y_{4(s-1)+1} = h_1 x_{4(s+1)} + h_2 x_{4s+3} + h_3 x_{4s+2} \\
y_{4(s-1)+2} = h_2 x_{4(s+1)} + h_3 x_{4s+3} \\
y_{4(s-1)+3} = h_3 x_{4(s+1)} \\
y_{4s} = 0 \\
\left.\begin{aligned}
y_{4s+1} &= h_0 x_{4s+1} + h_1 x_{4s} + h_2 x_{4s-1} + h_3 x_{4s-2} \\
y_{4s+2} &= h_0 x_{4s+2} + h_1 x_{4s+1} + h_2 x_{4s} + h_3 x_{4s-1} \\
y_{4s+3} &= h_0 x_{4s+3} + h_1 x_{4s+2} + h_2 x_{4s+1} + h_3 x_{4s} \\
y_{4(s+1)} &= h_0 x_{4(s+1)} + h_1 x_{4s+3} + h_2 x_{4s+2} + h_3 x_{4s+1}
\end{aligned}\right\} \text{Les } N \text{ sorties correspondant à une convolution linéaire après le recouvrement}
\end{cases}
$$

Donc, pour avoir une convolution linéaire classique, un recouvrement des sorties est nécessaire. Ce dernier est effectué à l'aide d'un délai de N échantillons comme le montre la figure (2.5).

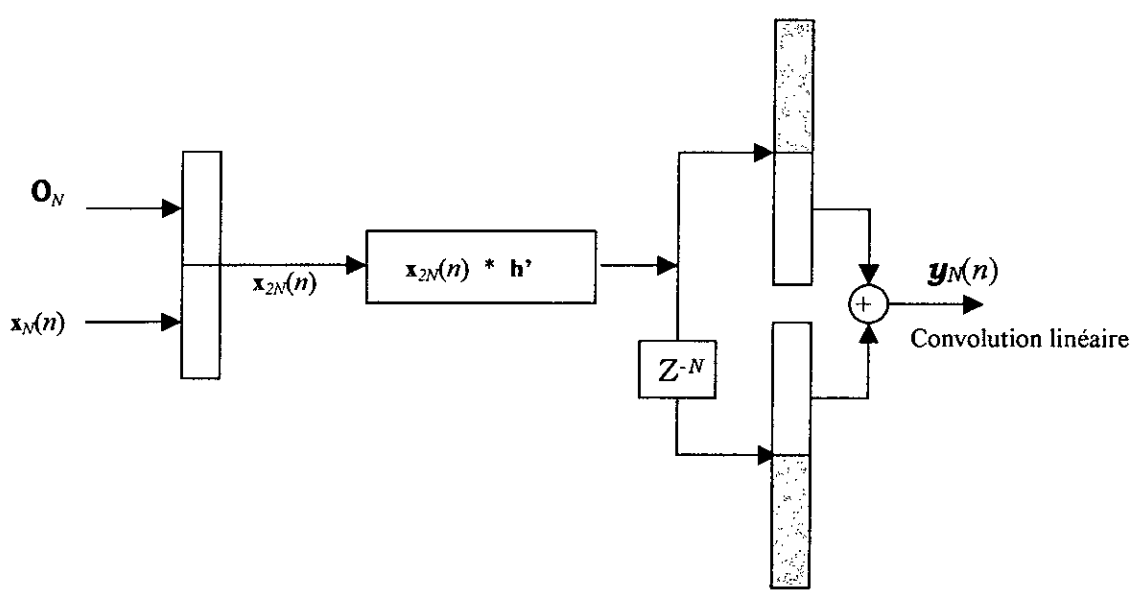

Figure 2.5 : Recouvrement des sorties

Dans le domaine fréquentiel, en utilisant la propriété de décomposition de la matrice circulante, l'algorithme qui en découle peut se résumer comme le montre la figure (2-6) :

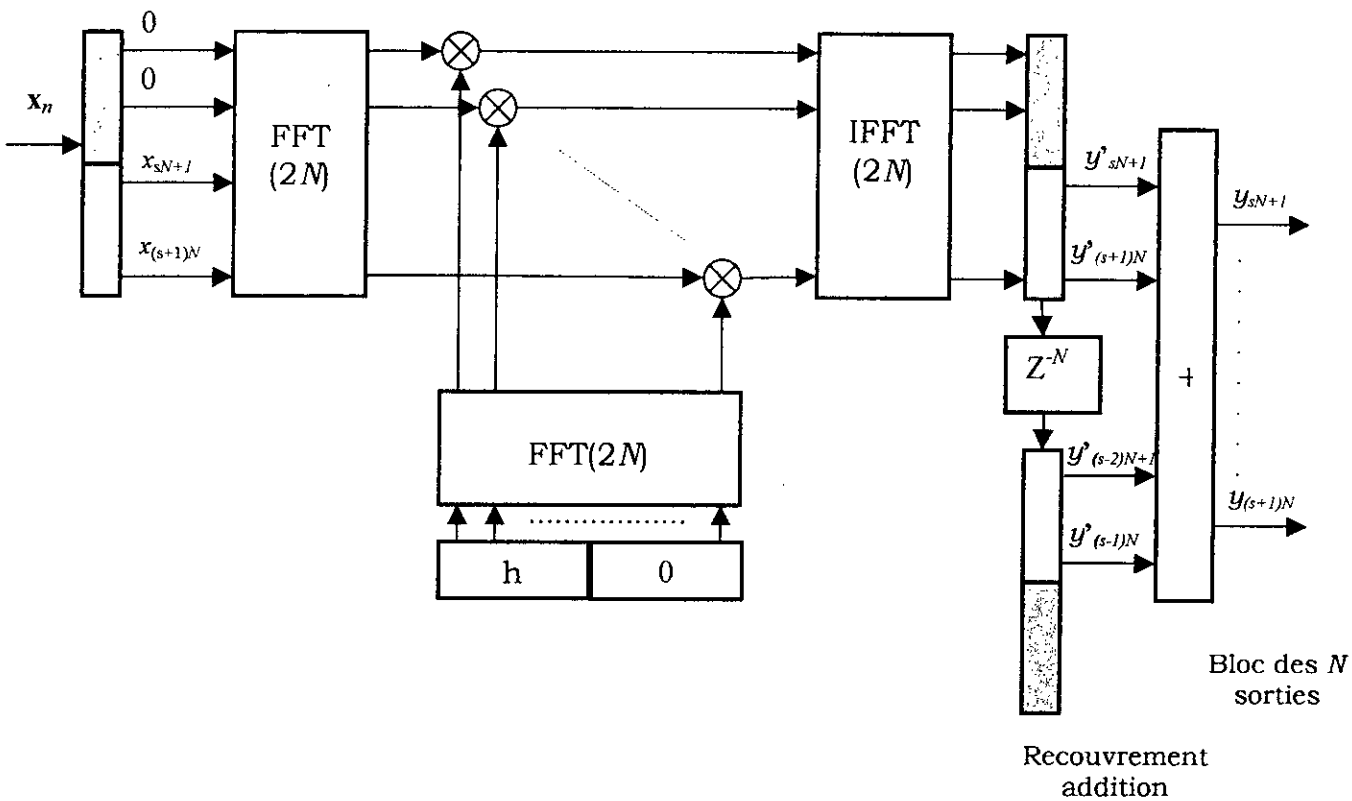

Figure 2.6 : Convolution rapide par OLA

Comme pour la technique OLS, l'organigramme suivant représente la convolution rapide par OLA dans le domaine fréquentiel figure (2.7) :

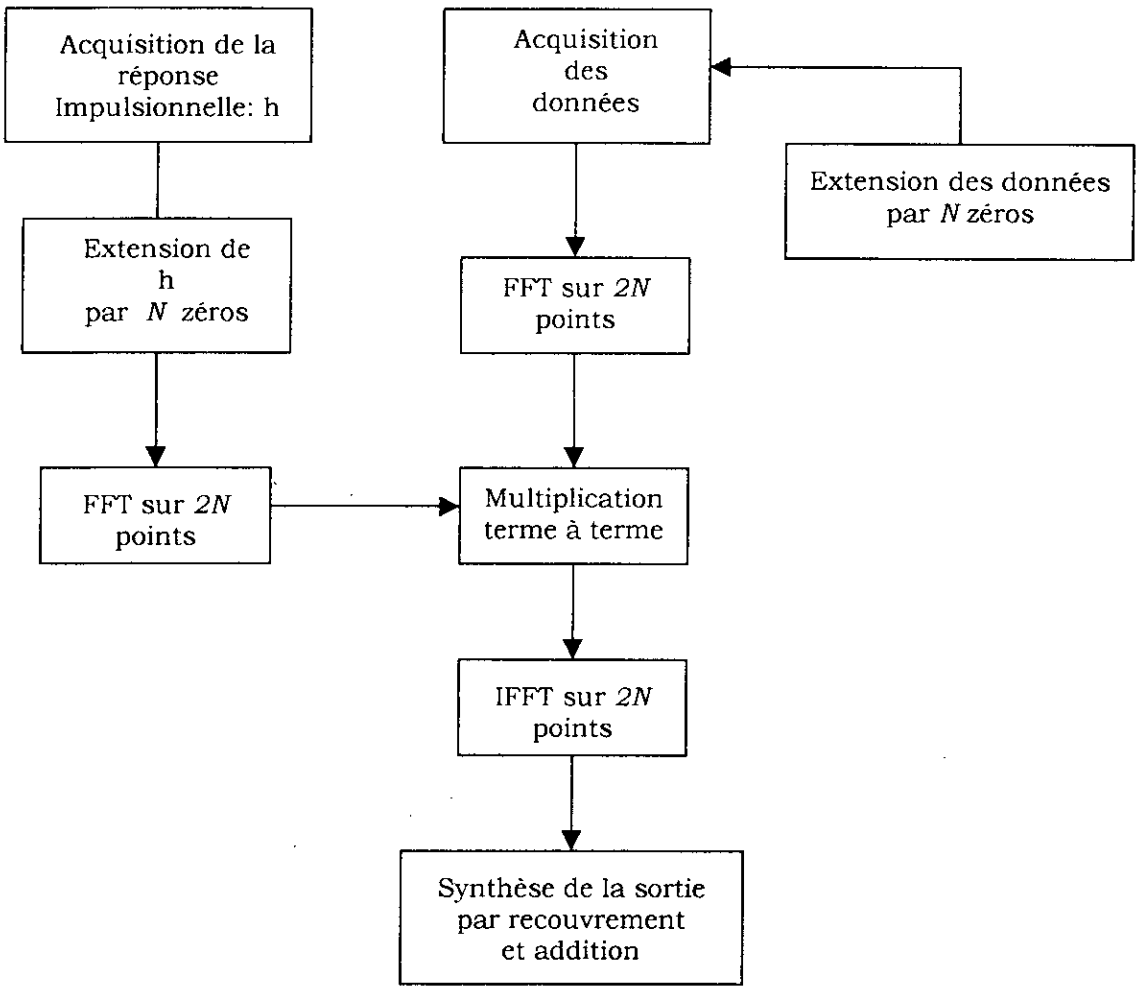

Figure 2.7 : Organigramme de la technique OLA

La technique OLS « over lap-save » découle de la sauvegarde du bloc passé pour ainsi former la convolution linéaire donnant les sorties y_{sN}. Tant dis que la technique OLA « over lap-add » est produite au moyen de blocs d'entrées disjoints des blocs de sorties par recouvrement et par addition. Par conséquent, ces deux techniques requièrent presque la même complexité.

De la même façon que pour la technique OLS, l'implémentation de la technique OLA en MATLAB donne des résultats identiques à ceux d'une convolution linéaire classique mais avec une réduction notable de la complexité arithmétique.

Dans le cas de la convolution discrète le nombre de multiplications est de $2N$ et celui d'additions $N(N\text{-}1)$, tandis que pour la convolution OLA, exploitant déjà la réduction de charge de calcul en utilisant la FFT, on peut améliorer le gain de façon plus significatif en utilisant l'algorithme FFT Split-Radix [DUH 86] de façon à avoir $[2N\log_2 N + 4]$ multiplications réelles et $[6N\log_2 N - N + 8]$ additions réelles avec $2N = 2^n$ (suite à une FFT de longueur $2N$, une FFT inverse de même longueur et de $2N$ multiplications complexes terme à terme et N additions dues au recouvrement.).

On remarque aisément, comme pour la technique OLS, que le gain en opérations est plus intéressant à mesure que le nombre d'échantillons augmente (figure 2.8).

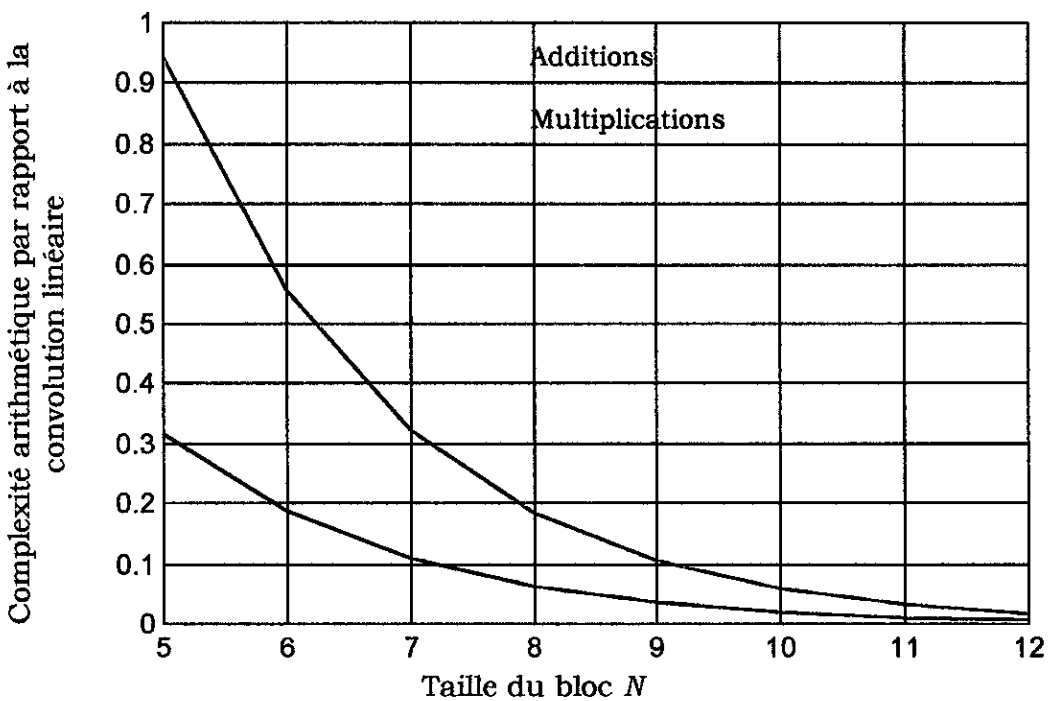

Figure 2.8 : Complexité par rapport à l'évaluation directe de la convolution

2.2.3 Convolution rapide par WOLA

Cette technique est une extension plus générale de la technique OLS. Les blocs d'entrées de taille $2N$ se recouvrent de la même manière que cette dernière tel que les $(N+1)$ points du bloc courant de sortie découle d'une convolution linéaire. Ces points sont alors isolés grâce à une fenêtre de troncature dont les propriétés sont comme suit :

$$\begin{cases} f_n \neq 0 & n \in [1,N] \\ f_n = 0 & \text{ailleurs} \end{cases} \qquad (2.12)$$

Donc, on définit une matrice de troncature :

$$\mathbf{f} = Diag[0, \ldots, 0, f_1, \ldots f_n] \qquad (2.13)$$

Soit α un entier tel que $\alpha \geq 1$ et $R=[N+1]/\alpha$, où [x] est la partie entière de x.

L'évaluation de la forme matricielle aux instants $n = sR$ se traduit par :

$$\boldsymbol{y'}_{sR}=\Psi_{sR}\mathbf{h'} \qquad (2.14)$$

le signal $\boldsymbol{z}_{sR} = \mathbf{f}\ \boldsymbol{y'}_{sR}$ de longueur $2N$ résultant du fenêtrage (des composantes non – nulles) découlent d'une convolution linéaire tel que :

$$\boldsymbol{z}_{sR} = \mathbf{f}\ \boldsymbol{y'}_{sR} = \mathbf{f}\ \Psi_{sR}\ \mathbf{h'} = \mathbf{f}\ \mathbf{W}^{-1}\ \mathbf{X}_{sR}\ \mathbf{W}\ \mathbf{h'} = \mathbf{f}\ \mathbf{W}^{-1}\ \mathbf{X}_{sR}\ \mathbf{H} \qquad (2.15)$$

tel que $z_{sR} = [\, z_{sR,0}, \ldots, z_{sR,2N-1}]$ où le premier indice représente l'origine des temps et le second celui de la position de l'échantillon au sein du bloc (mobile). Pour la formulation OLS le bloc des sorties successives ne se recouvrent pas, ce qui n'est pas le cas pour la technique WOLA où le même échantillon du signal peut être synthétisé à partir de α blocs adjacents obéissant à la formulation suivante :

$$y_n = \sum_s z_{sR,n} \tag{2.16}$$

Cette façon de synthétiser la sortie induit un recouvrement-addition sur la fenêtre elle même et impose alors la condition de normalisation :

$$\sum_s f_{n-sR} = 1 \tag{2.17}$$

La figure (2.9) donne une représentation explicite de cette technique :

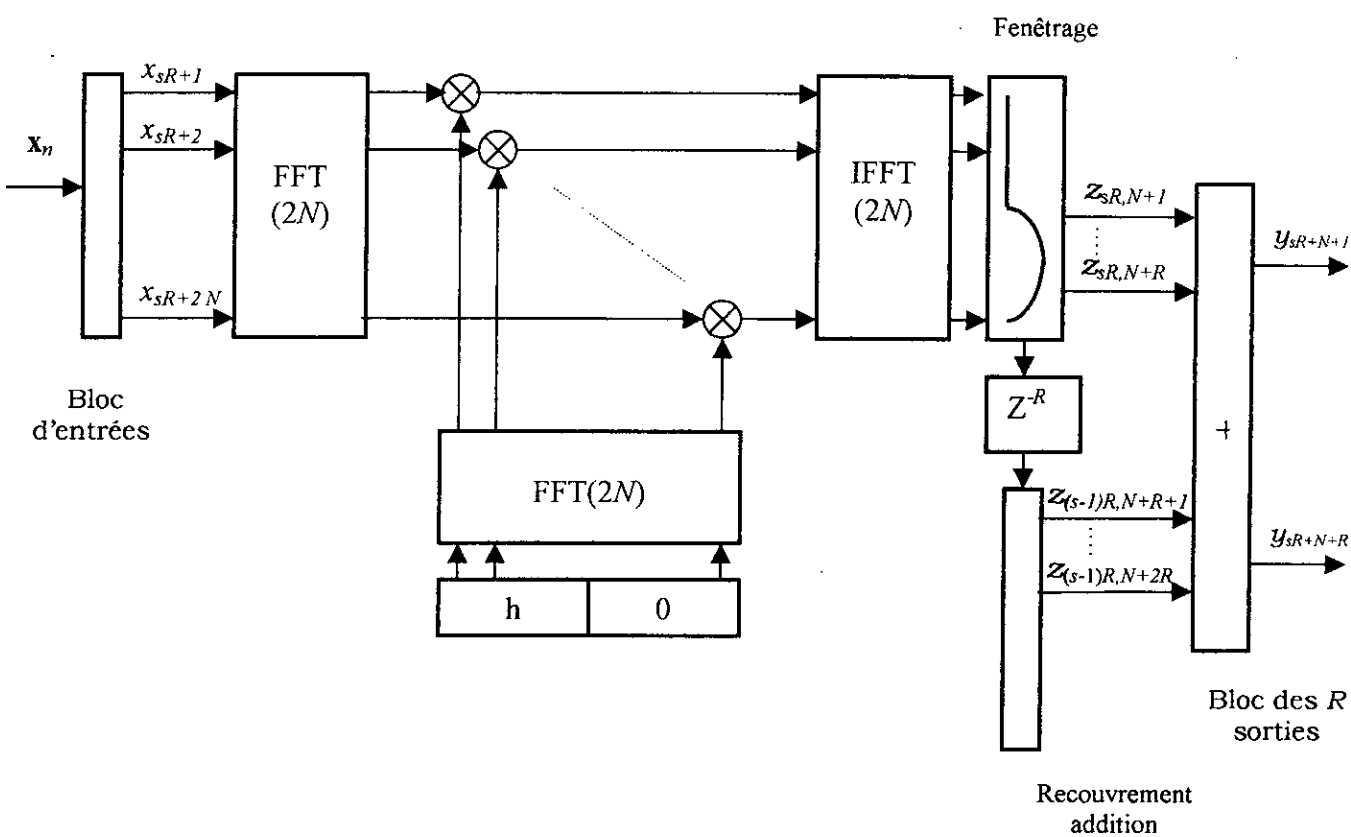

Figure 2.9 : Convolution rapide par WOLA

On remarquera que pour $\alpha = 1$, la condition (2.17) est réalisée par une fenêtre rectangulaire de longueur N ce qui conduit au cas particulier de la technique OLS.

L'exemple suivant permettra une meilleur compréhension de cette méthode :

On prendra $N = 8$ et $\alpha = 4$ ce qui donnera $R = 2$ et on fera une évaluation du bloc de sortie après chaque décalage (on prendra pour cet exemple quatre décalage temporel).

(sR)	$(s+1)R$	$(s+2)R$	$(s+3)R$
y'_1	y'_3	y'_5	y'_7
y'_2	y'_4	y'_6	y'_8
y'_3	y'_5	y'_7	y'_9
y'_4	y'_6	y'_8	y'_{10}
y'_5	y'_7	y'_9	y'_{11}
y'_6	y'_8	y'_{10}	y'_{12}
y'_7	y'_9	y'_{11}	y'_{13}
y'_8	y'_{10}	y'_{12}	y'_{14}
$f_1\ y'_9$	$f_1\ y'_{11}$	$f_1\ y'_{13}$	$f_1\ y'_{15}$
$f_2\ y'_{10}$	$f_2\ y'_{12}$	$f_2\ y'_{14}$	$f_2\ y'_{16}$
$f_3\ y'_{11}$	$f_3\ y'_{13}$	$f_3\ y'_{15}$	$f_3\ y'_{17}$
$f_4\ y'_{12}$	$f_4\ y'_{14}$	$f_4\ y'_{16}$	$f_4\ y'_{18}$
$f_5\ y'_{13}$	$f_5\ y'_{15}$	$f_5\ y'_{17}$	$f_5\ y'_{19}$
$f_6\ y'_{14}$	$f_6\ y'_{16}$	$f_6\ y'_{18}$	$f_6\ y'_{20}$
$f_7\ y'_{15}$	$f_7\ y'_{17}$	$f_7\ y'_{19}$	$f_7\ y'_{21}$
$f_8\ y'_{16}$	$f_8\ y'_{18}$	$f_8\ y'_{20}$	$f_8\ y'_{22}$

Dans notre exemple le bloc de sortie $(z_{(s+3)R,15}\ ,\ z_{(s+3)R,16})=(y_{15},y_{16})$ se répète α fois, donc d'après la relation (2.16)on aura :

$$\begin{cases} y_{15}= f_1\ y'_{15} + f_3\ y'_{15} + f_5\ y'_{15} + f_7\ y'_{15} \\ y_{16}= f_2\ y'_{16} + f_4\ y'_{16} + f_6\ y'_{16} + f_8\ y'_{16} \end{cases} \longrightarrow \begin{cases} f_1 + f_3 + f_5 + f_7 = 1 \\ f_2 + f_4 + f_6 + f_8 = 1 \end{cases}$$

Ce qui vérifie la relation (2.15). Et de façon plus générale, si son prend $y_{sR,n}$ comme bloc de sortie où le premier indice représente le temps d'évaluation et le second la position de l'échantillon dans ce bloc courant on a :

$$y_{sR,N+i} = f_{N+i-\alpha R}\, y'_{sR,N+i} + f_{N+i-(\alpha-1)R}\, y'_{(s-1)R,N+i+R} + \ldots$$

$$\ldots + f_{N+i-2R}\, y'_{(s-\alpha+2)R,(\alpha-2)R+N+i} + f_{N+i-R}\, y'_{(s-\alpha+1)R,(\alpha-1)R+N+i}$$

Tel que :

$i \in [1, R]$,

$N+ i = n$: représente le numéro de l'échantillon,

sR (tel que s $\in [1, \alpha]$) : représente le décalage temporel.

y'_i = les échantillons de sorties avant fenêtrage.

Donc, en éliminant les $y_{j,k}$ de l'équation précédente (ils sont tous égaux car il s'agit du même échantillon mais décalé dans le temps seulement), la formulation finale sera de la forme :

$$f_{n-\alpha R} + f_{n-(\alpha-1)R} + \ldots + f_{n-2R} + f_{n-R} = \Sigma f_{n-sR} = 1$$

L'organigramme de cette technique peut se résumer donc comme suit :

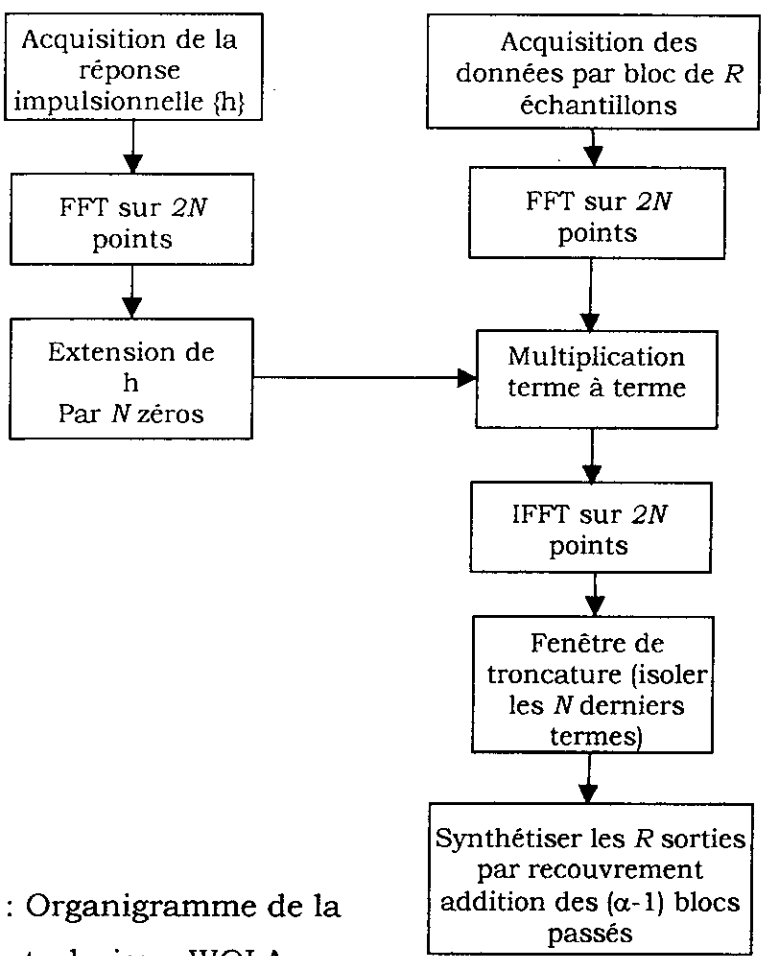

Figure 2.10 : Organigramme de la technique WOLA

Comme pour les deux techniques précédentes, les résultats obtenus en implémentant cet algorithme en MATLAB concordent exactement avec ceux d'une convolution linéaire classique.

En utilisant l'algorithme de la FFT Split-Radix [DUH 86] pour une réduction plus sensible de la complexité arithmétique, le nombre de multiplications réelles serait de $[2N\log_2 n + 4]$ et celui d'additions réelles de $[6N\log_2 N - (1+1/\alpha)\,N + 8]$ (figure 2.11).

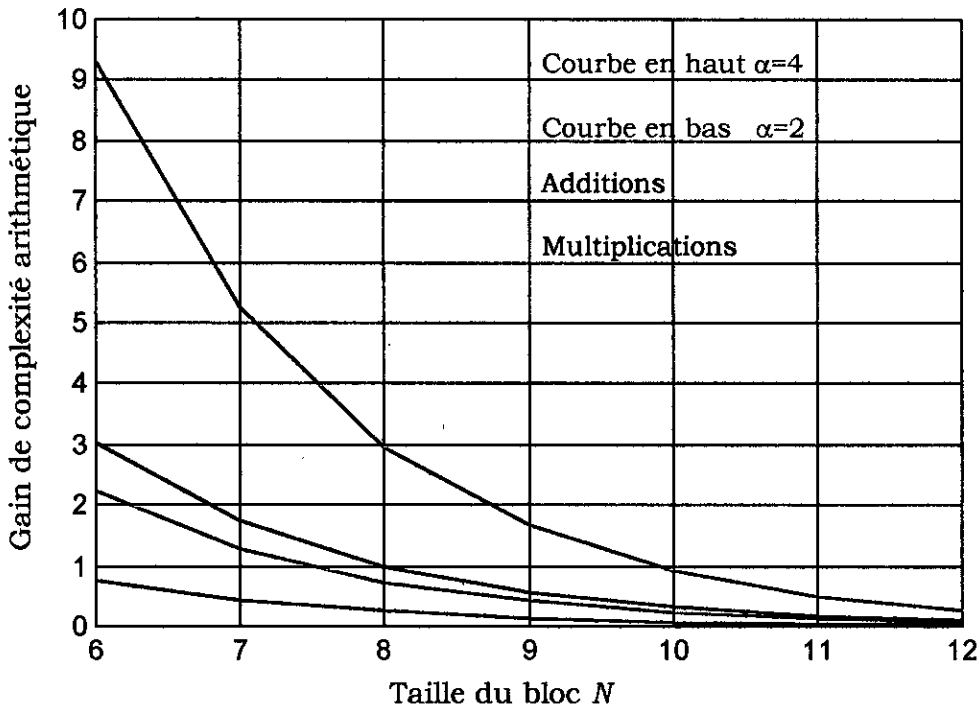

Figure 2.11 : Complexité par rapport à l'évaluation directe de la convolution

Cette technique ne présente pas d'intérêt dans le filtrage fixe, car le suréchantillonnage opéré engendre une charge de calcul croissant linéairement en α et nettement plus élevée que dans le cas des techniques OLS et OLA. Cependant, en filtrage adaptatif, l'augmentation de complexité est compensée par de meilleurs performances. En effet le suréchantillonnage conduit à une mise à jour plus fréquente du filtre adaptatif et améliore ainsi la vitesse de convergence et la capacité de poursuite.

2.2.4 Algorithmes à délai réduit

Dans les techniques de convolution rapide étudiées précédemment, l'acquisition des données était faite en fonction de la longueur de la réponse impulsionnelle en blocs de N échantillons. La causalité du filtre entraîne alors un délai de N échantillons dans le traitement des données. Ce délai devient un problème à résoudre car dans des applications en temps réel, où les réponses impulsionnelles mises en jeu sont longues (téléconférence par exemple), des délais prohibitifs sont inacceptables. Une segmentation de la réponse impulsionnelle en petits blocs permet alors de pallier cette limitation structurelle.

On obtient alors un outil de convolution rapide où le délai de convolution est ajusté par segmentation. A travers ce bref exposé, nous introduirons la procédure de segmentation pour la technique WOLA utilisée par la suite.

Considérons une réponse impulsionnelle de longueur L segmentée en K blocs de longueur N tel que $L=KN$:

$$\mathbf{h} = [h_0, h_1, \ldots\ldots, h_{N-1}, \ldots\ldots, h_{2N-1}, \ldots\ldots, h_{L-1}]^T$$

Après segmentation nous aurons :

$$\mathbf{h} = [[h_0, \ldots\ldots, h_{N-1}][\, h_N, \ldots\ldots, h_{2N-1}] \ldots\ldots\ldots\ldots [h_{L-N}, \ldots\ldots, h_{L-1}]]^T$$

$$= [\mathbf{h}^{(0)} \;\; \mathbf{h}^{(1)} \;\; \mathbf{h}^{(2)} \ldots\ldots\ldots\ldots\ldots \mathbf{h}^{(K-2)} \;\; \mathbf{h}^{(K-1)}]^T$$

De la même façon pour les blocs d'entrées :

$$\mathbf{x}_n = [x_n, x_{n-1}, \ldots\ldots, x_{n-N+1}, \ldots\ldots, x_{n-2N+1}, \ldots\ldots, x_{n-L+1}]^T$$

$$= [[x_n, \ldots\ldots, x_{n-N+1}][x_{n-N}, \ldots\ldots, x_{n-2N+1}] \ldots\ldots [\, x_{n-N-L}, \ldots\ldots, x_{n-L+1}]]^T$$

$$= [\; \mathbf{x}_n^{(0)} \;\; \mathbf{x}_n^{(1)} \;\; \ldots\ldots\ldots\ldots \mathbf{x}_n^{(K-2)} \; \mathbf{x}_n^{(K-1)}]^T$$

la convolution à l'instant n s'écrit alors :

$$\begin{aligned} y_n = {} & x_n h_0 + x_{n-1} h_1 + \ldots\ldots\ldots\ldots + x_{n-N+1} h_{N-1} + \\ & x_{n-N} h_N + x_{n-N-1} h_{N+1} + \ldots\ldots\ldots + x_{n-2N+1} h_{2N-1} + \\ & \ldots\ldots \quad \ldots\ldots \quad \ldots\ldots \\ & \ldots\ldots \quad \ldots\ldots \quad \ldots\ldots \\ & x_{n-N-L} h_{L-N} + x_{n-L-N-1} h_{L-N+1} + \ldots\ldots\ldots\ldots + x_{n-L+1} h_{L-1} \end{aligned}$$

En utilisant la forme vectorielle on peut écrire :

$$y_n = \mathbf{x}_n^{(0)} \mathbf{h}^{(0)T} + \mathbf{x}_n^{(1)} \mathbf{h}^{(1)T} + \ldots\ldots\ldots\ldots + \mathbf{x}_n^{(K-2)} \mathbf{h}^{(K-2)T} + \mathbf{x}_n^{(K-1)} \mathbf{h}^{(K-1)T}$$

Donc, et de façon similaire à (2.4), la sortie du filtre à l'instant n s'écrira :

$$y_n = \sum_{k=0}^{K-1} \mathbf{h}^{(k)} \mathbf{x}_n^{(k)} \qquad (2.18)$$

Avec $\mathbf{h}^{(k)} = [h_{kN}, \ldots, h_{kN+N-1}]^T$

$$\mathbf{x}_n^{(k)} = [x_{n-kN}, \ldots, x_{n-kN-N+1}]^T$$

En, introduisant la matrice circulante et de façon similaire aux équations (2.14 – 2.15) avec $n=sR$, le bloc courant s'écrit :

$$\mathbf{y}'_{sR} = \sum_{k=0}^{K-1} \Psi_{sR-kN} \mathbf{h}'^{(k)} \qquad (2.19)$$

La propriété de la matrice circulante induisant une décomposition dans la base de Fourier, nous permet d'écrire la relation (2.19) :

$$\mathbf{y}'_{sR} = \mathbf{W}^{-1} \sum_{k=0}^{K-1} \mathbf{X}_{sR-kN} \mathbf{H}^{(k)} \qquad (2.20)$$

où $\mathbf{H}^{(k)}$ n'est autre que la TFD du vecteur $\mathbf{h}^{(k)}$ complété par N zéros et $\mathbf{X}_{sR-kN}$ la matrice diagonale de dimension $2N$ contenant les coefficients de la TFD de la première colonne de Ψ_{sR-kN}.

On remarque alors que le filtrage dans le domaine transformé n'est réalisé qu'à travers des produits complexes (simples) terme à terme. En effet, la relation (2.20) montre clairement que la segmentation fait apparaître des convolutions dont la longueur est égale au nombre de segments (figure 2.12).

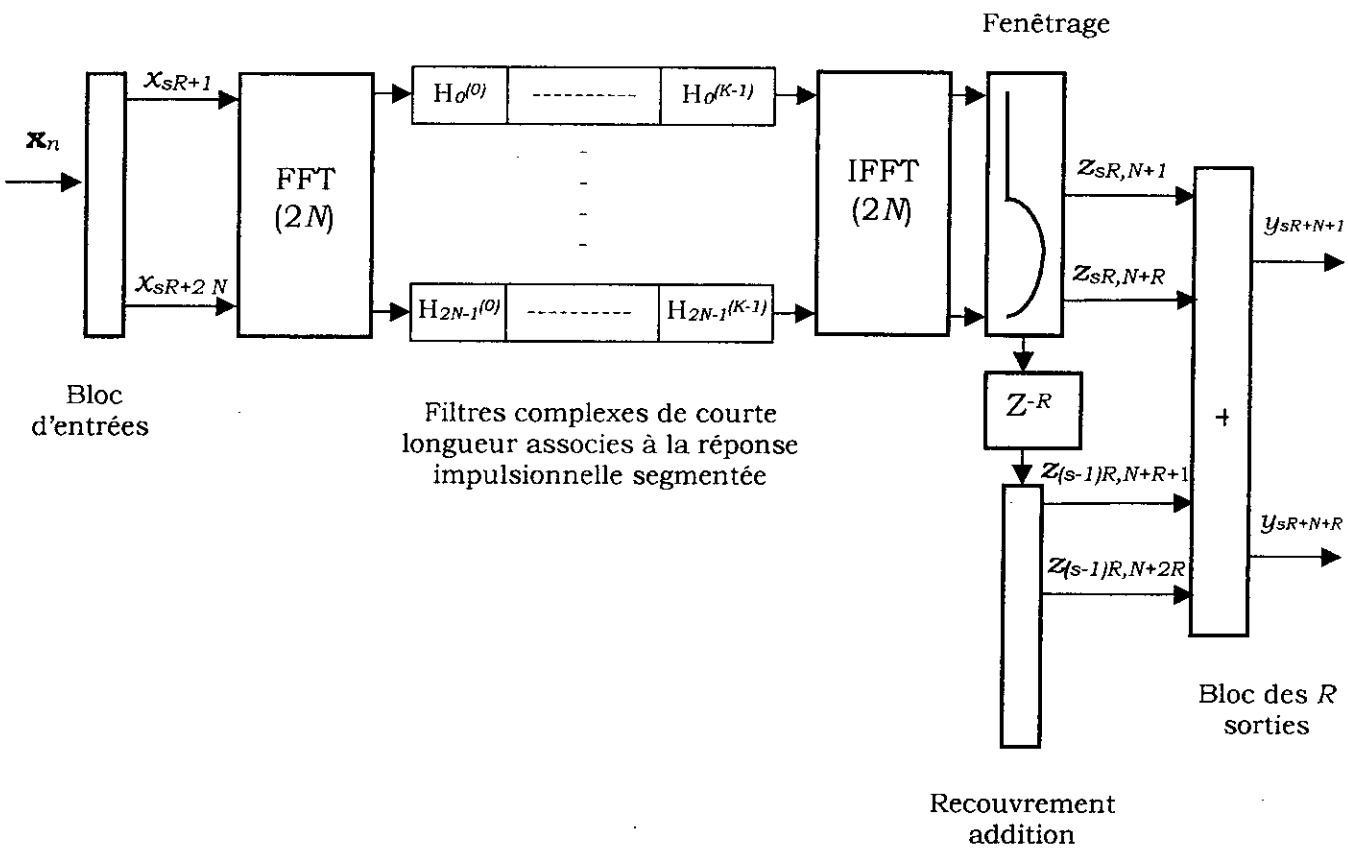

Figure 2.12 : Convolution rapide par WOLA et segmentation

De la même manière que pour la technique WOLA, les N composantes du bloc courant de $2N$ sorties découlant d'une convolution linéaire sont isolés au

moyen d'une fenêtre de troncature définie en (2.12). le bloc de sortie s'écrit alors :

$$\boldsymbol{f}\, \boldsymbol{y}'_{sR} = \boldsymbol{f} \sum_{k=0}^{K-1} \Psi_{sR-kN} \mathbf{h}^{(k)} = \boldsymbol{f}\, \mathbf{W}^{-1} \sum_{k=0}^{K-1} \mathbf{X}_{sR-kN} \mathbf{H}^{(k)} \qquad (2.21)$$

La sortie est synthétisée de façon similaire à (2.16) par recouvrement – addition des signaux courts termes successifs z_{sR} et impose de même la condition de normalisation (2.17).

L'organigramme de cette méthode peut alors se résumer comme suit :

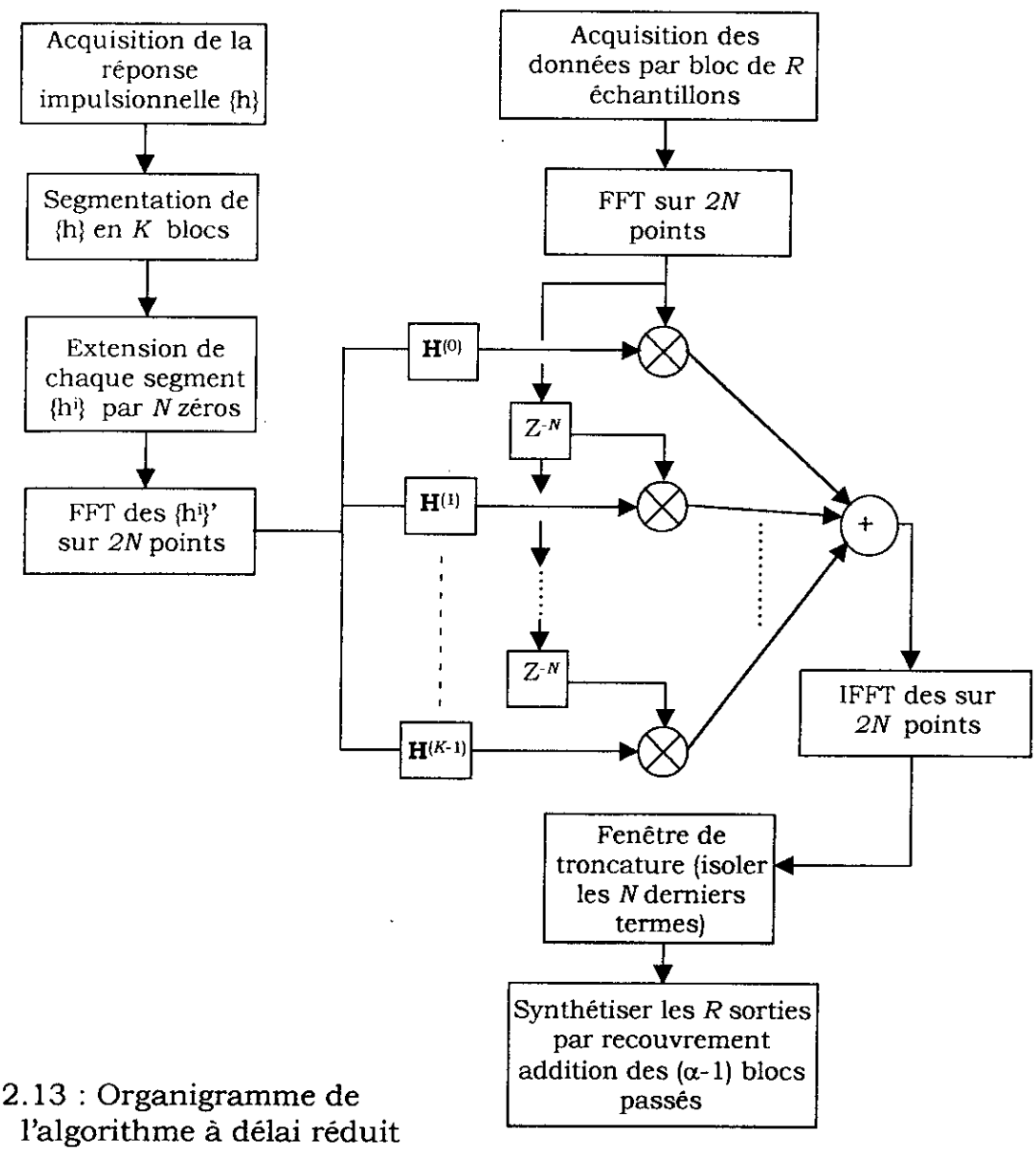

Figure 2.13 : Organigramme de l'algorithme à délai réduit

Le filtrage de R échantillons au moyen de l'algorithme FFT Split-Radix [DUH 86] nécessite $2N\ log_2N + 4N(K-1) + 4$ multiplications réelles et $6N\ log_2N + 2N(2K - 3) + 8$ (figure (2.17)). On remarque clairement, pour cette méthode, que la complexité arithmétique croit en fonction du nombre de segments K (figure 2.14).

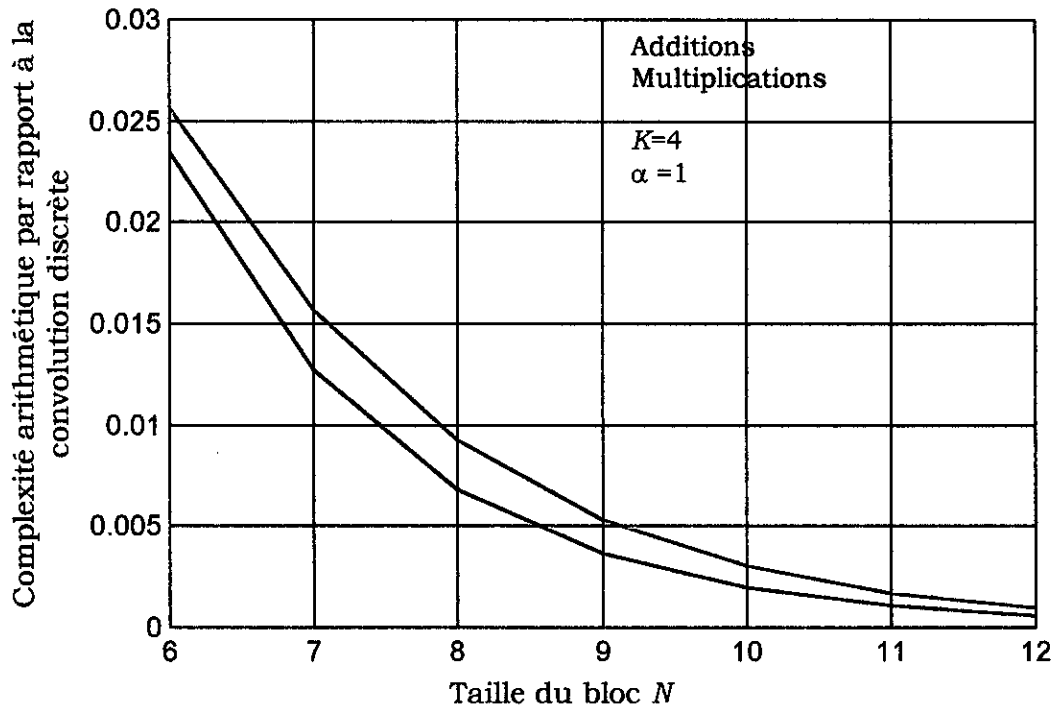

Figure 2.14 : Complexité par rapport à l'évaluation directe de la convolution

Chapitre 3

Algorithme de filtrage adaptatif dans le domaine fréquentiel

Le filtrage adaptatif est défini par deux processus (figure 3.1) :

- La convolution, qui est une partie préliminaire et très importante, permet une première construction des filtres adaptatifs. Ainsi qu'il a été montré dans le chapitre précédent, l'étude des différentes techniques de convolution dans le domaine fréquentiel nous a permis d'identifier les problèmes dus à la complexité arithmétique (charge de calcul) ainsi que le délai prohibitif de traitement. D'où la nécessité de promouvoir une structure par bloc (segmentation de la réponse impulsionnelle) qui arrive à palier ces problèmes.

- La mise à jour des coefficients de la réponse impulsionnelle (partie adaptatif, qui fait l'objet de ce chapitre) permet d'évaluer et de façon efficace les variations temporelles de ces derniers. Notons que dans notre cas, la partie adaptatif se fera également dans la domaine fréquentiel

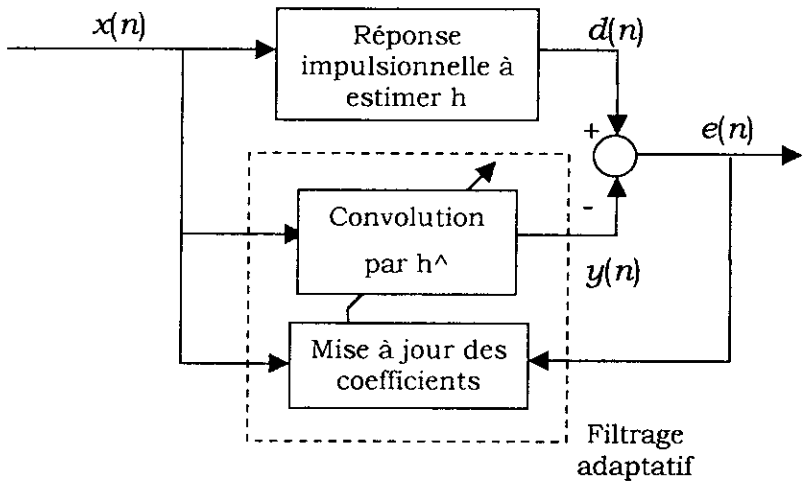

Figure 3.1 : Filtrage adaptatif

Dans ce processus de filtrage adaptatif (partie adaptation), est utilisée une procédure appelée procédure de gradient qui sert à minimiser l'énergie du signal d'erreur, de façon à estimer les coefficients de la réponse impulsionnelle le plus précisément possible. Cette procédure de gradient peut être aisément modifiée de façon à utiliser un pas d'adaptation approprié pour chaque coefficient du filtre.

En effet, la vitesse de convergence de l'algorithme du gradient est inversement proportionnelle à le dispersion des valeurs propres de la matrice d'autocorrélation du signal d'entrée. Ces valeurs propres correspondent approximativement au spectre de puissance discret du signal d'entrée. Il est donc possible de compenser cette dispersion en utilisant pour chaque coefficient du filtre un pas d'adaptation inversement proportionnel à la puissance du signal d'excitation correspondant ($\mu_N=\mu/|x(n)|^2$) [HAY 91]. En conséquence, une convergence quasi-uniforme des divers modes du processus est réalisée.

Après ce bref exposé sur le filtrage adaptatif, nous introduirons, en première partie de ce chapitre, les équations dans le domaine fréquentielles régissant l'algorithme GMDFα (objet de notre rapport), puis en deuxième partie l'établissement de cet algorithme ainsi que l'étude de la complexité arithmétique induite.

3.1 Filtrage adaptatif dans le domaine fréquentiel « Algorithme GMDFα »

D'après les techniques étudiées dans le chapitre précédent, nous avons constaté que, pour des applications qui nécessitent une assez longue réponse impulsionnelle tel l'annulation d'écho acoustique, un retard de traitement prohibitif est introduit. De plus le filtre étant mis à jour par bloc ne « voit » pas les variations rapides du système à identifier, d'où une capacité de poursuite dégradée. La segmentation de la réponse impulsionnelle du filtre et le suréchantillonnage des blocs (en utilisant la technique WOLA qui permet de calculer la convolution à une cadence α fois plus grande) permettent de s'affranchir de ces limitations au prix d'une moindre réduction de la complexité de calcul qui reste cependant nettement inférieure à celle de l'algorithme temporel.

Cette technique de suréchantillonnage intégrée dans l'algorithme fréquentiel par petits blocs (algorithme à délai réduit comme décrit dans le chapitre précédent) appliquée au filtrage adaptatif sera dénommée Generalized Multi-Delay Adaptive Filter α (GMDFα).

3.1.1 Equations de convolution

Dans le chapitre précédent, nous avons traité une procédure de convolution rapide (algorithme à délai réduit) qui minimise la contrainte de délai

par segmentation de la réponse impulsionnelle. Avant d'établir l'algorithme de mise à jour du filtre, nous rappelons brièvement les équations traduisant le filtrage dans le domaine transformé.

Avec une réponse impulsionnelle le longueur L ($L=K\,N$), et se référant à la dernière technique étudiée dans le chapitre précédent, la sortie y_n du filtre à l'instant n s'écrit :

$$y_n = \sum_{k=0}^{K-1} \mathbf{h}^{(k)} \mathbf{x}_n^{(k)} \tag{3.1}$$

avec $\mathbf{h}^{(k)} = [h_{kN}{}^T, \ldots, h_{kN+N-1}{}^T]^T$

$\mathbf{x}_n^{(k)} = [x_{n-kN}, \ldots, x_{n-kN-N+1}]^T$

L'équation (3.1) évaluée aux instant $n = sR$ ($R=[N+1]/\alpha$) :

$$\mathbf{y_{sR}} = \sum_{k=0}^{K-1} \Psi_{sR-kN} \mathbf{h}'^{(k)} = \mathbf{W}^{-1} \sum_{k=0}^{K-1} \mathbf{X}_{sR-kN} \mathbf{H}^{(k)} \tag{3.2}$$

avec

$\mathbf{H}^{(k)} = \mathbf{W}\,[\mathbf{h}^{(k)T}, \mathbf{O}_N{}^T]$.

$\mathbf{X}_{sR-kN} = Diag\,[X_{sR-kN}, \ldots\ldots, X_{sR-kN+2N-1}]$

Cette représentation dans le domaine transformé peut être réécrite de façon plus compacte :

$$\mathbf{y}_{sR} = \mathbf{W}^{-1}\mathbf{X}(sR)\mathbf{H} \tag{3.3}$$

avec

$\mathbf{H} = [\mathbf{H}^{(0)T}, \ldots\ldots, \mathbf{H}^{(K-1)T}]$ «vecteur de dimension $2KN$».

$\mathbf{X}(sR) = [\mathbf{X}_{sR}, \ldots\ldots, \mathbf{X}_{sR-(K-1)N}]$ «matrice $2N \times 2KN$».

Le suréchantillonnage inhérent à la technique WOLA introduit alors une fenêtre de troncature qui isole les N dernières sorties qui découlent d'une convolution linéaire :

$$\mathbf{Z}_{sR} = \mathbf{f}\,\mathbf{W}^{-1}\,\mathbf{X}(sR)\,\mathbf{H} \tag{3.4}$$

Le recouvrement-addition de la sortie sur α blocs donne :

$$y_n = \sum_s z_{sR,n} \tag{3.5}$$

3.1.2 Equations de mise à jour des coefficients de la réponse impulsionnelle

Comme on impose que les coefficients du filtre restent constants sur la durée du bloc, les calculs peuvent être réalisés dans le domaine de Fourier de façon équivalente à la forme temporelle. Cette forme fréquentielle est plus efficace car elle met en œuvre la convolution rapide.

Considérons la matrice circulante Ψ_n définie précédemment (dans le chapitre 2) pour $n \in [n - N,\ n + N - 1]$, l'équation de convolution par bloc s'écrit :

$$\boldsymbol{y}_n = \Psi_n \mathbf{h'}_n \tag{3.7}$$

Avec $\boldsymbol{y}_n = [y_{n-N}, \ldots\ldots, y_{n-1}, y_n, \ldots\ldots, y_{n+N-1}]^T$ dont seules les N derniers correspondent à une convolution linéaire classique.

$$\mathbf{h'}_n = [h_0, h_1, \ldots\ldots, h_{N-1}, 0, 0, \ldots\ldots, 0, 0]^T$$

En projetant l'équation (3.7) sur la base de Fourier, on trouve :

$$\mathbf{W}\,\boldsymbol{y}_n = \mathbf{W}\,\Psi_n\,\mathbf{W}^{-1}\,\mathbf{W}\;\mathbf{h'}_n = \mathbf{X}_n\,\mathbf{W}\,\mathbf{h'}_n \tag{3.8}$$

En appliquant la relation d'adaptation par bloc (chapitre 1, équation (1.8))[GIL 92] écrite à l'instant $(n + N)$, avec $\mathbf{x}_n=\mathbf{Q}\Psi^{\mathbf{T}}n$ on trouve :

$$\mathbf{h'}_{n+N} = \mathbf{h'}_n + (\mu/2N)\ \mathbf{Q}\ \Psi^T_n\ \mathbf{e'}_n \qquad (3.9)$$

où $\mathbf{e'}_n = [\ \mathbf{0}_N\ \ \mathbf{e}_n]^T$, et $\mathbf{Q} = Diag\ [\mathbf{1}_N, \mathbf{0}_N]$ « fenêtre isolant les termes de corrections ». μ : le pas d'adaptation.

En appliquant l'équation (3.9) sur un filtre dont la longueur est $L=KN$ (segmenté en K blocs de longueur N), l'équation de mise à jour du $i^{ème}$ bloc s'écrit :

$$\mathbf{h'}^{(i)}_{n+N} = \mathbf{h'}^{(i)}_n + (\mu/2N)\ \mathbf{Q}\ \Psi^T_{n-iN}\ \mathbf{e'}_n \qquad (3.10)$$

En passant dans le domaine de Fourier :

$$\mathbf{W}\ \mathbf{h'}^{(i)}_{n+N} = \mathbf{W}\ \mathbf{h'}^{(i)}_n + (\mu/2N)\ \mathbf{W}\ \mathbf{Q}\ \mathbf{W}^{-1}\ \mathbf{X}^T_{n-iN}\ \mathbf{W}\ \mathbf{e'}_n$$

donc ;

$$\mathbf{H}^{(i)}_{n+N} = \mathbf{H}^{(i)}_n + (\mu/2N)\ \Pi\ \mathbf{X}^T_{n-iN}\ \mathbf{W}\ \mathbf{e'}_n \qquad (3.11)$$

où $\mathbf{H}^{(i)}_{n+N} = \mathbf{W}\ \mathbf{h'}^{(i)}_{n+N} = [h^{(i)}_0, h^{(i)}_1, \ldots\ldots, h^{(i)}_{N-1}, 0, 0, \ldots\ldots, 0, 0]^T$ est un vecteur de dimension $2N$, et $i \in [0, K-1]$ représentant le numéro du bloc.

$\Pi = Diag\ [\mathbf{W}\ \mathbf{Q}\ \mathbf{W}^{-1}, \ldots\ldots, \mathbf{W}\ \mathbf{Q}\ \mathbf{W}^{-1}]$ est un projecteur ;

avec $$\mathbf{Q} = Diag[\mathbf{1}_N\ \mathbf{0}_N] = \begin{pmatrix} \mathbf{I}_N & \mathbf{0}_N \\ \mathbf{0}_N & \mathbf{0}_N \end{pmatrix} \qquad (3.12)$$

Dans l'algorithme décrit ci-dessus, le filtre est rafraîchi une fois tous les N échantillons, ce qui limite la capacité de poursuite. On s'affranchit de cette limitation en utilisant une technique de type recouvrement – addition pondérée

(Weighted Over Lap-Add : WOLA) qui permet de calculer la convolution à une cadence α fois plus grande. En posant $R=[(N+1)/\alpha]$, on effectue les calculs (convolution et adaptation) tous les R échantillons.

Donc, pour des systèmes à réponse impulsionnelle longue, en utilisant la technique WOLA et en posant $n=sR$; (3.11) s'écrit :

$$\mathbf{H}(s+1) = \mathbf{H}(s) + (\mu/2N)\ \Pi\ \mathbf{X}^T(sR)\ \mathbf{E}_{sR} \qquad (3.13)$$

Cette formulation peut être généralisée en substituant au projecteur défini en (3.12) l'opérateur Γ déduit d'une fenêtre de troncature adaptée au contexte d'utilisation du filtre. Donc, on peut écrire :

$$\mathbf{H}(s+1) = \mathbf{H}(s) + (\mu/2N)\ \ \Gamma\ \mathbf{X}^T(sR)\ \mathbf{E}_{sR} \qquad (3.14)$$

avec :

- $\mathbf{H} = [\mathbf{H}^{(0)T}, \ldots\ldots, \mathbf{H}^{(K-1)T}]$ «vecteur de dimension $2KN$».
- $\mathbf{X}(sR) = [\mathbf{X}_{sR}, \ldots\ldots, \mathbf{X}_{sR-(K-1)N}]$ «matrice $2N \times 2KN$».
- $\mathbf{E}_{sR} = \mathbf{W}\ \mathbf{f}\ [\ \mathbf{d}_{sR} - \mathbf{W}^{-1}\ \mathbf{X}(sR)\ \mathbf{H}(s)] = \mathbf{F}\ [\ \mathbf{D}_{sR} - \mathbf{X}(sR)\ \mathbf{H}(s)]$
- $\mathbf{F} = \mathbf{W}\ \mathbf{f}\ \mathbf{W}^{-1}$ et $\mathbf{f} = Diag\ [0, \ldots\ldots, 0, f_1, \ldots\ldots, f_N]$.
- $\nabla(sR) = \mathbf{X}^T(sR)\ \mathbf{E}_{sR}$: Gradient « terme correcteur ».

Nous remarquons d'après la relation (3.14) que chaque coefficient de la réponse impulsionnelle (dans le domaine de Fourier) est adaptée indépendamment des autres. Donc on peut régler l'adaptation individuellement de façon à accélérer la convergence en multipliant le terme correctif $\mathbf{X}^T(sR)\ \mathbf{E}_{sR}$

par une matrice diagonale $\mathbf{T}(s)$ de dimension $2N$ mise à jour à chaque itération. L'équation (3.14) devient alors :

$$\mathbf{H}(s+1) = \mathbf{H}(s) + (\mu/2N)\ \Gamma\ \mathbf{T}(s)\ \mathbf{X}^T(sR)\ \mathbf{E}_{sR} \qquad (3.15)$$

En choisissant $\mathbf{T}(s)$ comme la matrice inverse de la matrice de puissance d'entrée (diagonale) ($T_{kk}(s) = [P_k(s)]^{-1}$) le pas d'adaptation va être normalisé de façon à éviter la dispersion des valeurs propres de la matrice d'autocorrélation du bloc d'entrée et ainsi favoriser la convergence quasi-uniforme des différents coefficients du filtre.

On choisira comme estimateur récursif de la matrice de puissance :

$$P_k(s+1) = \gamma\ P_k(s) + (1-\gamma)\,|\mathbf{X}_{sR,k}|^2 \qquad (3.16)$$

où γ est le facteur d'oubli.

Donc, l'algorithme général est donné par l'équation (3.15) à travers laquelle nous pouvons développer deux structures du GMDFα :

- Si $\Gamma = \mathbf{I}$, l'algorithme sera dénommé GMDFα non contraint; dans lequel des termes de repliement du produit $\mathbf{X}^T(sR)\mathbf{E}_{sR}$ vont s'ajouter.

- Si $\Gamma = \Pi$, l'algorithme sera dénommé GMDFα contraint dans lequel les termes de repliement sont éliminés grâce au projecteur Π.

Pour mieux illustrer ces deux points, nous prendrons un exemple avec N=4, K=1, et α=1, (R=N).

Donc l'équation (3.15) devient :

$$\mathbf{H}(s+1) = \mathbf{H}(s) + (\mu/2N)\ \Gamma\ \mathbf{T}(s)\ \mathbf{X}^T(sN)\ \mathbf{E}_{sN} \qquad (3.17)$$

avec $\mathbf{X}(sN) = Diag\,[\,\mathrm{FFT}(x_{sN}\ x_{sN+1}\ x_{sN+2}\ x_{sN+3}\ x_{sN+4}\ x_{sN+5}\ x_{sN+6}\ x_{sN+7})]$

$\mathbf{E}_{sN} = \mathrm{FFT}\,[0\ 0\ 0\ 0\ e_{sN+4}\ e_{sN+5}\ e_{sN+6}\ e_{sN+7}]$

$\mathbf{T}(s) = Diag\,[t_0\ t_1\ t_2\ t_3\ t_4\ t_5\ t_6\ t_7]$

1- Dans le cas où $\Gamma = \mathbf{I}$: l'équation précédente devient :

$$\mathbf{H}(s+1) = \mathbf{H}(s) + (\mu/2N)\,\mathbf{T}(s)\,\mathbf{X}^T(sN)\,\mathbf{E}_{sN} \qquad (3.18)$$

posons : $\Phi = \mathbf{T}(s)\,\mathbf{X}^T(sN)\,\mathbf{E}_{sN} = [\phi_0\ \phi_1\ \phi_2\ \phi_3\ \phi_4\ \phi_5\ \phi_6\ \phi_7]^T$

donc, l'équation (3.18) s'écrira :

$$\begin{pmatrix} H_0(s+1) \\ H_1(s+1) \\ H_2(s+1) \\ H_3(s+1) \\ H_4(s+1) \\ H_5(s+1) \\ H_6(s+1) \\ H_7(s+1) \end{pmatrix} = \begin{pmatrix} H_0(s) \\ H_1(s) \\ H_2(s) \\ H_3(s) \\ H_4(s) \\ H_5(s) \\ H_6(s) \\ H_7(s) \end{pmatrix} + (\mu/2N) \begin{pmatrix} \phi_0 \\ \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \\ \phi_6 \\ \phi_7 \end{pmatrix}$$

On remarque qu'au passage dans domaine temporel, les N derniers termes dus au repliement $[\phi_4\ \phi_5\ \phi_6\ \phi_7]$ vont introduire des erreurs sur les coefficients réels de la réponse impulsionnelle $[h_0(s+1)\ h_1(s+1)\ h_2(s+1)\ h_3(s+1)]$ à cause de la transformée inverse de Fourier qui est sur $2N$ points et qui inclue ces termes de repliements.

2- Dans le cas où $\Gamma = \Pi$, l'équation (3.17) devient :

$$\mathbf{H}(s+1) = \mathbf{H}(s) + (\mu/2N)\, \Pi\, \mathbf{T}(s)\, \mathbf{X}^T(sN)\, \mathbf{E}_{sN} \qquad (3.19)$$

En passant dans le domaine temporel, en mettant $\Pi = Diag\,[\mathbf{W}\,\mathbf{Q}\,\mathbf{W}^{-1},\ldots\ldots,\,\mathbf{W}\,\mathbf{Q}\,\mathbf{W}^{-1}]$ (avec $\mathbf{Q} = Diag\,[\mathbf{1}_N\ \mathbf{0}_N]$) et $\Phi = \mathbf{T}(s)\,\mathbf{X}^T(sN)\,\mathbf{E}_{sN} = [\phi_0\ \phi_1\ \phi_2\ \phi_3\ \phi_4\ \phi_5\ \phi_6\ \phi_7]^T$, on trouve :

$$\mathbf{h}(s+1) = \mathbf{h}(s) + (\mu/2N)\, \mathbf{Q}\, \mathbf{W}^{-1}\, \varphi \qquad (3.20)$$

avec $\varphi = \mathbf{W}^{-1}\, \Phi$.

La forme vectorielle (plus explicite) nous donne :

$$\begin{pmatrix} h_0(s+1) \\ h_1(s+1) \\ h_2(s+1) \\ h_3(s+1) \\ h_4(s+1) \\ h_5(s+1) \\ h_6(s+1) \\ h_7(s+1) \end{pmatrix} = \begin{pmatrix} h_0(s) \\ h_1(s) \\ h_2(s) \\ h_3(s) \\ h_4(s) \\ h_5(s) \\ h_6(s) \\ h_7(s) \end{pmatrix} + (\mu/2N) \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \varphi_0 \\ \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \varphi_5 \\ \varphi_6 \\ \varphi_7 \end{pmatrix}$$

Nous remarquons qu'avec l'utilisation du projecteur, que les termes de repliements sont éliminés et de ce fait que les N derniers coefficients sont nuls, sans qu'il y ait d'erreurs altérants l'adaptation des coefficients réels de la réponse impulsionnelle $[h_0(s+1)\ h_1(s+1)\ h_2(s+1)\ h_3(s+1)]$.

Remarque : La différence entre ces deux structures du GMDFα sera étudiée dans le chapitre Simulation et résultats, mais nous pouvons déjà remarquer qu'en contre partie des erreurs induites dans le cas de l'algorithme

non contraint, on constate une complexité arithmétique plus accrue dans le cas de l'algorithme contraint (à cause du projecteur).

On peut donc définir la structure générale de l'algorithme GMDFα (figure (3.2)) où le signal d'entrée est transformé au moyen du banc de filtre TFD. Le filtrage est effectué de façon efficace par des convolutions de courte longueur entre les *2N* signaux issus du banc de filtre TFD et les filtres complexes de tailles *K*. La sortie estimée est évaluée au moyen du banc de synthèse par TFD inverse et pondération (**f**). Le vecteur d'erreurs déduit de la transformation de l'erreur d'estimation *e*(*n*) par le banc d'analyse est alors fourni à la procédure qui calcul le terme correctif (gradient). Enfin, les incréments issus de cette procédure, après contrainte au moyen du projecteur Γ, ajustent les coefficients du filtre adaptatif.

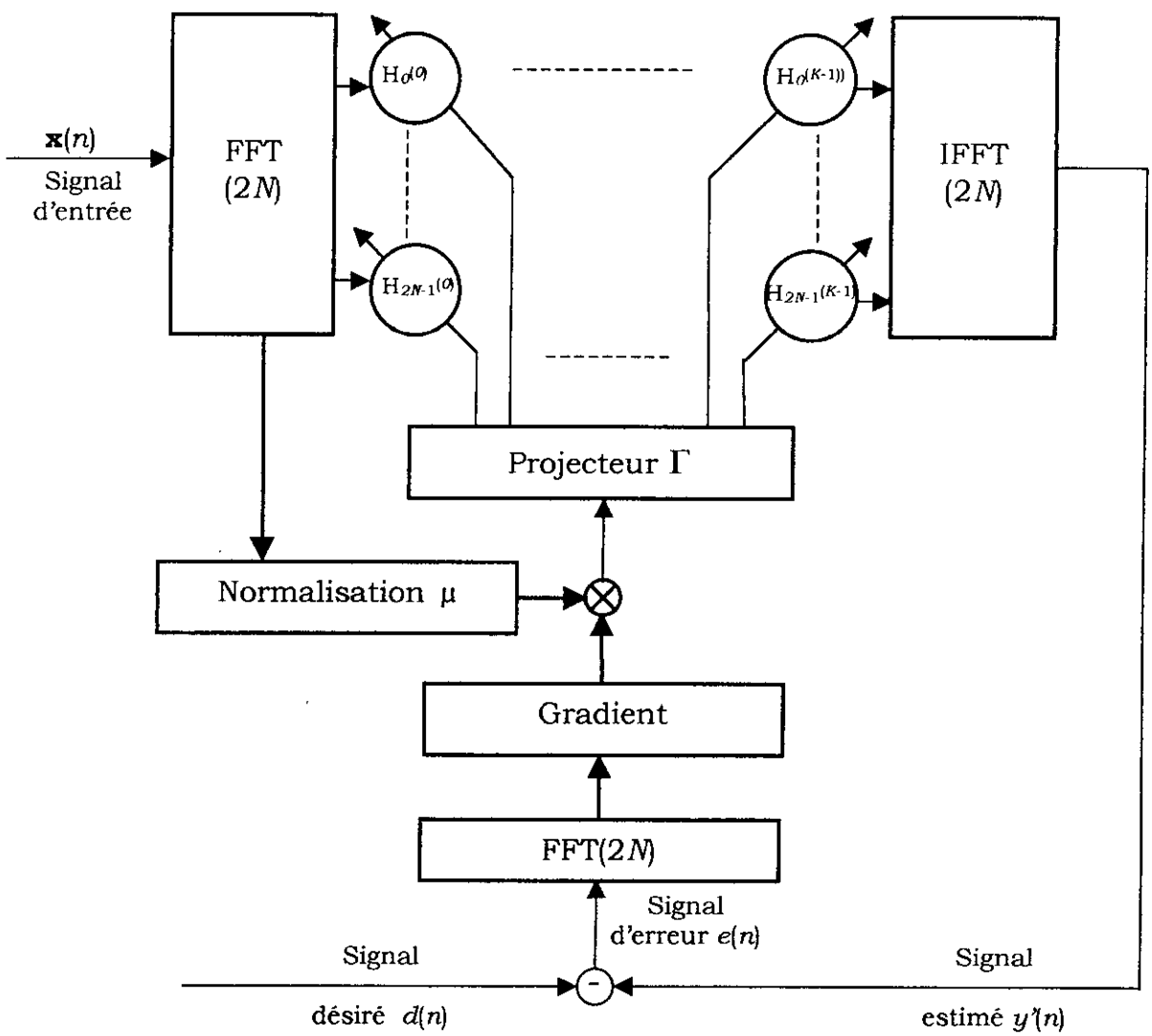

Figure 3.2: structure de l'algorithme GMDFα

Donc, l'organigramme principale de cet algorithme se résume comme suit :

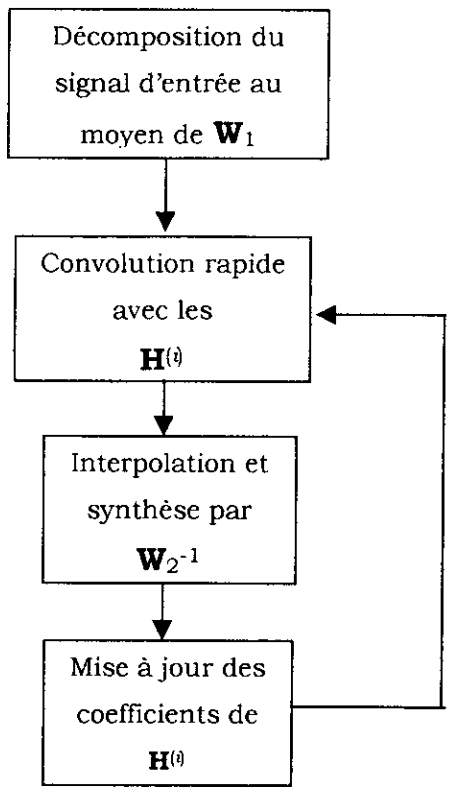

Figure 3.3 : Organigramme du GMDFα

3.2 Implantation des équations de filtrage

A travers l'étude des équations régissant l'algorithme GMDFα, les deux équations principales sont celle de l'estimation de l'erreur à l'aide de convolution rapide et celle de mise à jour des coefficients du filtre :

$$\begin{cases} \mathbf{H}^{(i)}(s+1) = \mathbf{H}^{(i)}(s) + (\mu/2N)\ \Gamma\ \mathbf{T}(s)\ \mathbf{X}^{T}_{sR\text{-}iN}\ \mathbf{E}_{sR} \\ \mathbf{E}_{sR} = \mathbf{W}\ \mathbf{f}[\mathbf{d}_{sR} - \mathbf{W}^{-1}\ \mathbf{X}(sR)\ \mathbf{H}(s)] \end{cases}$$

Les différentes étapes de cet algorithme consistent alors (figure 3.4) tous les R échantillons à :

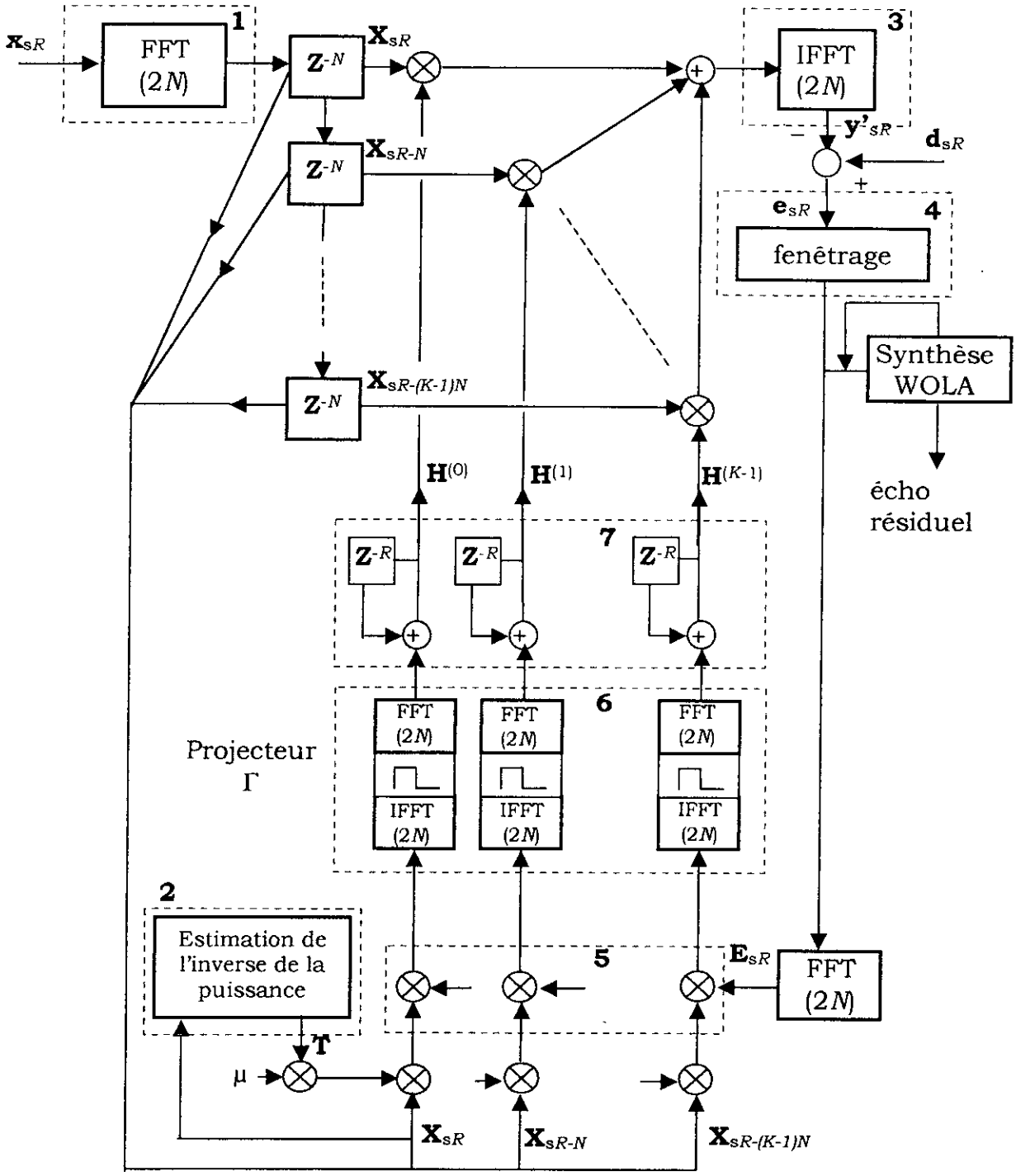

Figure 3.4 Structure détaillée du GMDFα

1 – Evaluation de la FFT du bloc d'entrées courant : $\mathbf{X}_{sR}$.

2 – Mise à jour des coefficients de normalisation : $T_{kk}(s) = [P_k(s)]^{-1}$.

3 – Evaluation du bloc de sorties temporelles estimées :

$$\boldsymbol{y}_{sR} = \mathbf{W}^{-1} \sum_{k=0}^{K-1} \mathbf{X}_{sR-kN} \mathbf{H}^{(k)}$$

4 – Evaluation du signal d'erreur : $\boldsymbol{e}_{sR} = \mathbf{f}[\, \mathbf{d}_{sR} - \boldsymbol{y}'_{sR}\,]$.

5 – Evaluation du terme correctif du $i^{ème}$ segment du filtre :

$$\Delta^{(i)}(s+1) = (\mu/2N)\, \mathbf{T}(s)\, \mathbf{X}^{T}_{sR-iN}\, \mathbf{E}_{sR}\,.$$

6 – Contrainte (optionnelle) : $\Delta^{(i)}(s+1) = \Gamma\, \Delta^{(i)}(s+1)$.

7 – Mise à jour du $i^{ème}$ segment du filtre : $\mathbf{H}^{(i)}(s+1) = \mathbf{H}^{(i)}(s) + \Delta^{(i)}(s+1)$.

3.3 Complexité arithmétique du GMDFα

Nous évaluons la complexité arithmétique de l'algorithme GMDFα (figure 3.5)[AIT 92] par rapport à celle du NLMS qui comprend $(2N+5)$ multiplications réelles et $(2N+1)$ additions réelles « pour des blocs de $N=2^P$ ». Le nombre d'opérations nécessaire au calcul d'une FFT correspond à l'algorithme FFT Split-Radix [DUH 86].

	Multiplications réelles / Echantillons	Additions réelles/ échantillons
Non contraint	$\alpha[10K + 3b - 3]$	$\alpha[8K + 9b - 10]$
Contraint	$\alpha[(2K + 3)b + 4K - 3]$	$\alpha[(6K + 9)b - 2K - 10]$

Figure 3.5 : Complexité arithmétique du GMDFα

La comparaison de la complexité arithmétique par rapport au NLMS sera effectuée pour les deux structures du GMDFα : contraint et non contraint (multiplications :figure (3.6), additions :figure (3.7)).

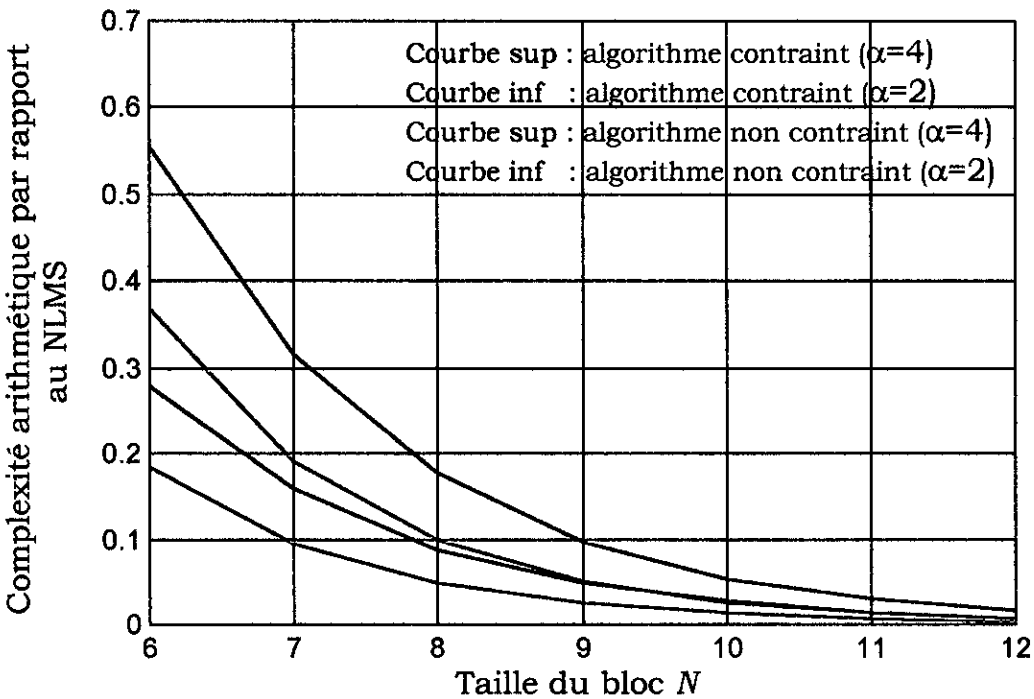

Figure 3.6 : Complexité du GMDFα par rapport au NLMS (multiplications)

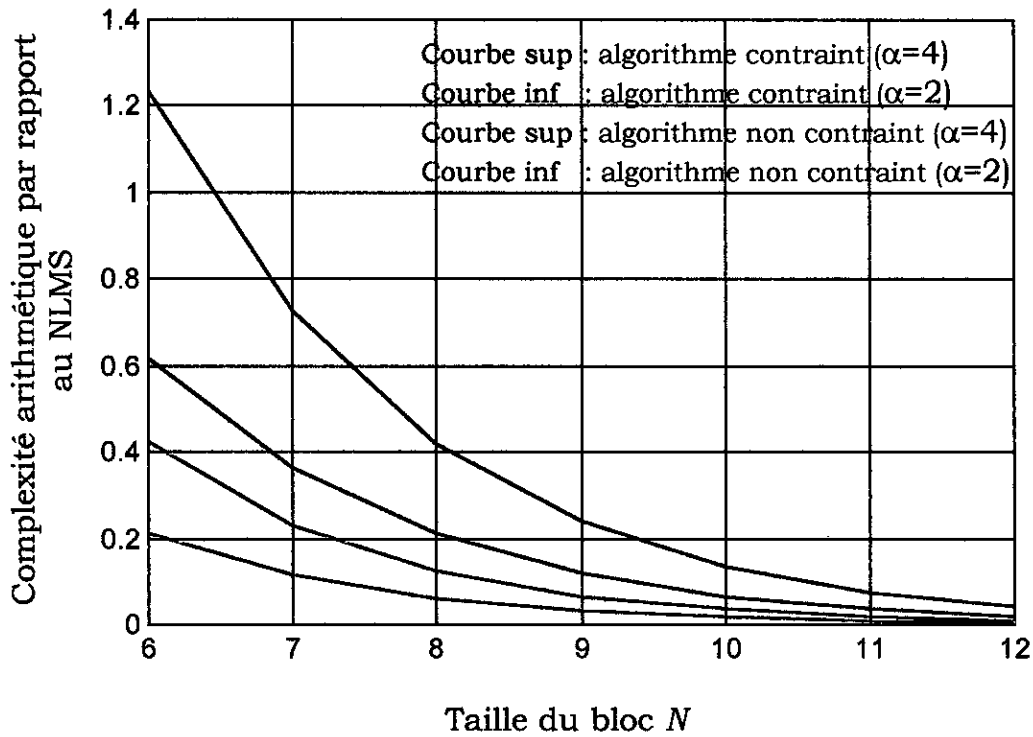

Figure 3.7 : Complexité du GMDFα par rapport au NLMS (additions)

Chapitre 4
Simulation et résultats

Nous avons développé précédemment les équations associées à l'algorithme GMDFα et les étapes qui permettent l'implémentation d'un filtre adaptatif. L'étude de ce chapitre se décompose en deux partie :

- Application du GMDFα pour l'identification de réponses impulsionnelles finies utilisant comme signaux d'entrée, des signaux aléatoires blancs.
- Application de l'algorithme GMDFα pour l'annulation d'écho d'un signal de parole.

4.1 Etude des performances du GMDFα

Dans cette étude, nous nous basons sur le cas particulier où le signal d'entrée $\mathbf{x}(n)$ - filtré par une réponse impulsionnelle finie - est une séquence aléatoires uniformément distribuée. La réponse impulsionnelle sera de la forme [SOO-PAN 90] :

$$h(i) = r(i) \exp(-0.04i) \qquad i = 1,2,...L.$$

où $r(i)$ est une variable aléatoire uniformément distribuée dans l'intervalle [0,1], et L la longueur du filtre.

La figure (4.1) montre le modèle de la simulation :

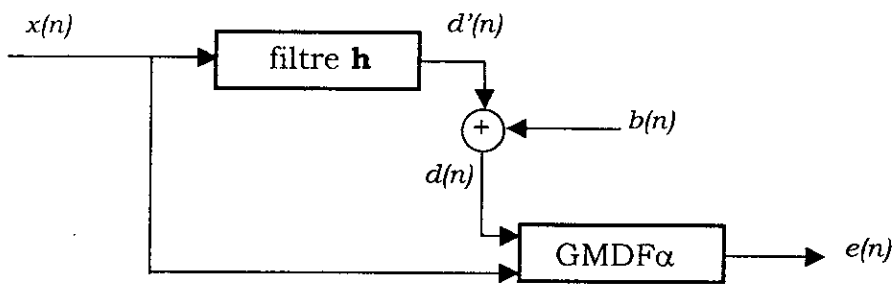

Figure 4.1 : Modèle de la simulation

Nous utiliserons l'erreur quadratique moyenne comme paramètre d'évaluation définie comme suit :

$$\mathrm{EQM}(s) = \frac{1}{N}\sum_{i=1}^{N} e_i^2$$

où e_i sont les éléments du vecteur contenant les N erreurs du bloc courant s.

Dans un premier temps, nous supposons que le signal $y(n)$ est nul, donc la réponse désiré sera la convolution de la séquence $\mathbf{x}(n)$ par le filtre **h**. Le pas d'adaptation μ sera choisit comme suit :

$$0 < \mu < 1/\lambda_{max}$$

où λ_{max} est la valeur maximale des valeurs propres de la matrice d'autocorrélation du bloc des entrées. Comme le montre la figure (4.2), l'algorithme GMDFα converge plus rapidement que le LMS.(pour un canal de longueur L=16).

Nous comparons aussi les performances de l'algorithme GMDFα (pour une réponse impulsionnelle de longueur L=16) avec celles du BLMS (figure 4.3) qui n'utilise pas la propriété de normalisation du pas d'adaptation par la puissance du bloc des entrées. Par conséquent, pour le BLMS, μ étant fixe le long du bloc, la convergence de cet algorithme sera plus faible que celle du GMDFα.

Le suréchantillonnage dans le GMDFα, permet de diminuer le délai d'évaluation en réduisant le nombre d'échantillons à traiter dans chaque bloc. En conséquence, la vitesse de convergence est plus importante pour un facteur de suréchantillonnage plus grand introduisant une baisse de précision tolérable par rapport à un facteur moins grand (figure 4.4). Cette baisse est due à un moyennage des (α-1) sorties précédantes après la procédure recouvrement – addition.

Comme cité dans le chapitre 3, le GMDFα possède deux structures : Contraint et Non Contraint. L'algorithme contraint, utilisant le projecteur, converge moins rapidement mais avec une précision plus importante que celle du non contraint. En effet, ce dernier prend en considération les N derniers termes de la réponse impulsionnelle dans le domaine transformé (longueur $2N$), entraînant des erreurs de repliement.

Le choix d'un algorithme de ce type est utile pour des applications où la précision importe peu et la charge de calcul est limitée. Deux exemples (figure 4.5 et 4.6) illustrent cette comparaison pour des segmentations différentes K=2 et K=4.

Nous supposant maintenant que la réponse désiré *d(n)* est perturbé par un bruit *b(n)*. L'ajout de ce bruit réduit le niveau de convergence d'autant plus que le rapport signal sur bruit (RSB) est faible (figure 4.7).

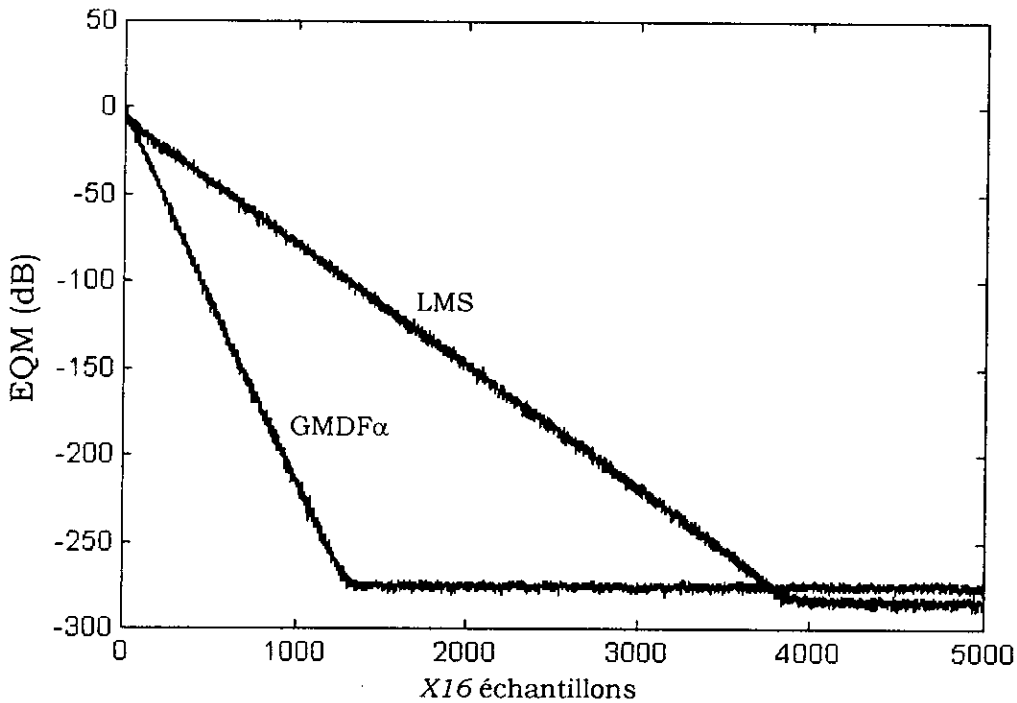

Figure 4.2 : Comparaison entre le GMDF et le LMS pour N=16, K=1, α=1

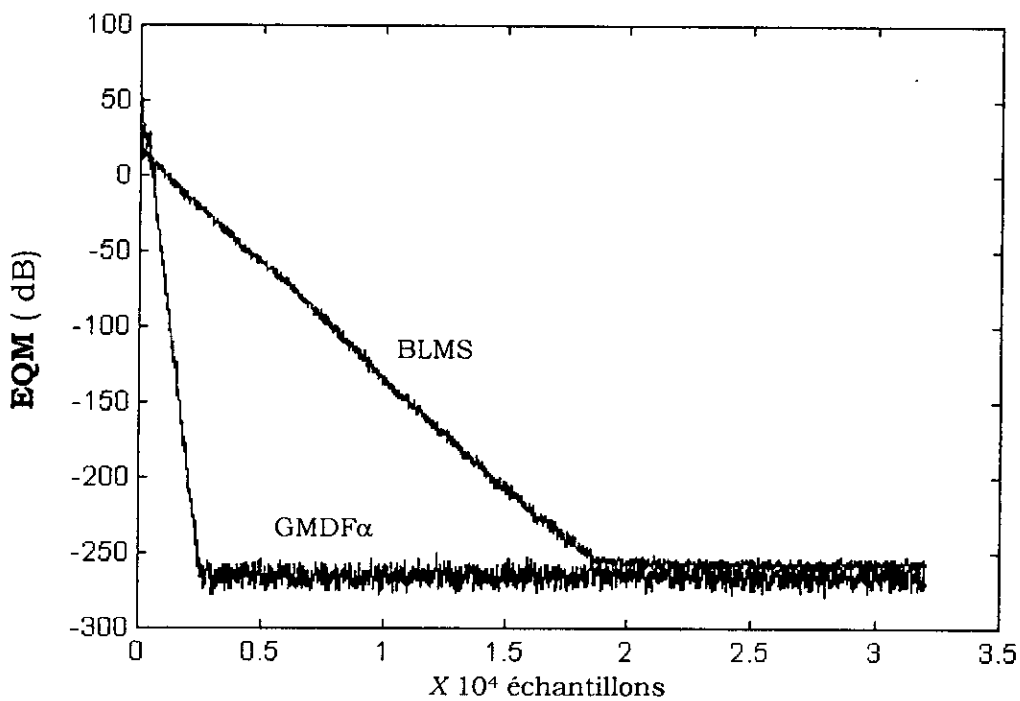

Figure 4.3 : Comparaison entre le GMDFα et le BLMS, pour $N = 16$, $K = 1$, $\alpha=1$

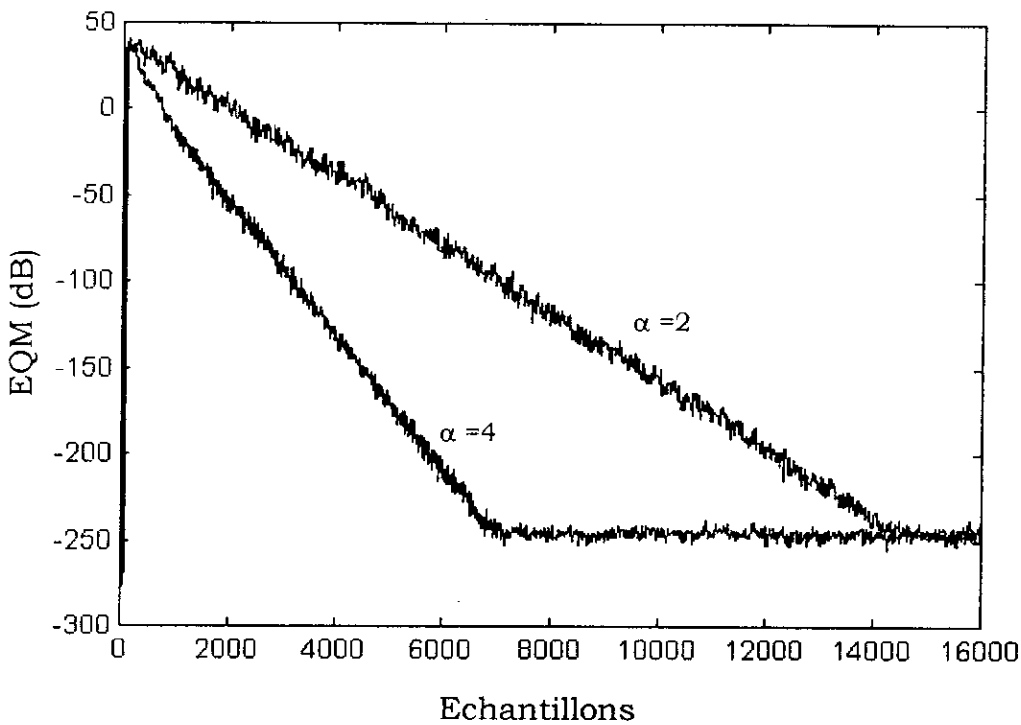

Figure 4.4 : Convergence du GMDFα α=2 et α=4, *N*=16, *K*=1

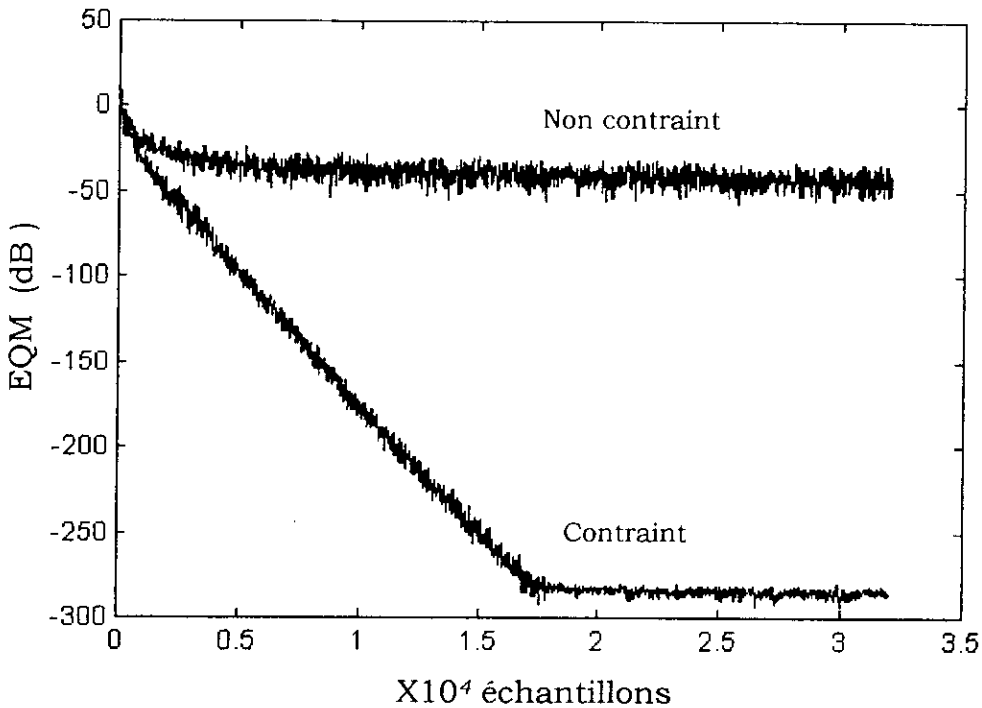

Figure 4.5 : Algorithme GMDFα contraint et non contraint pour K=2, N=16, α =1

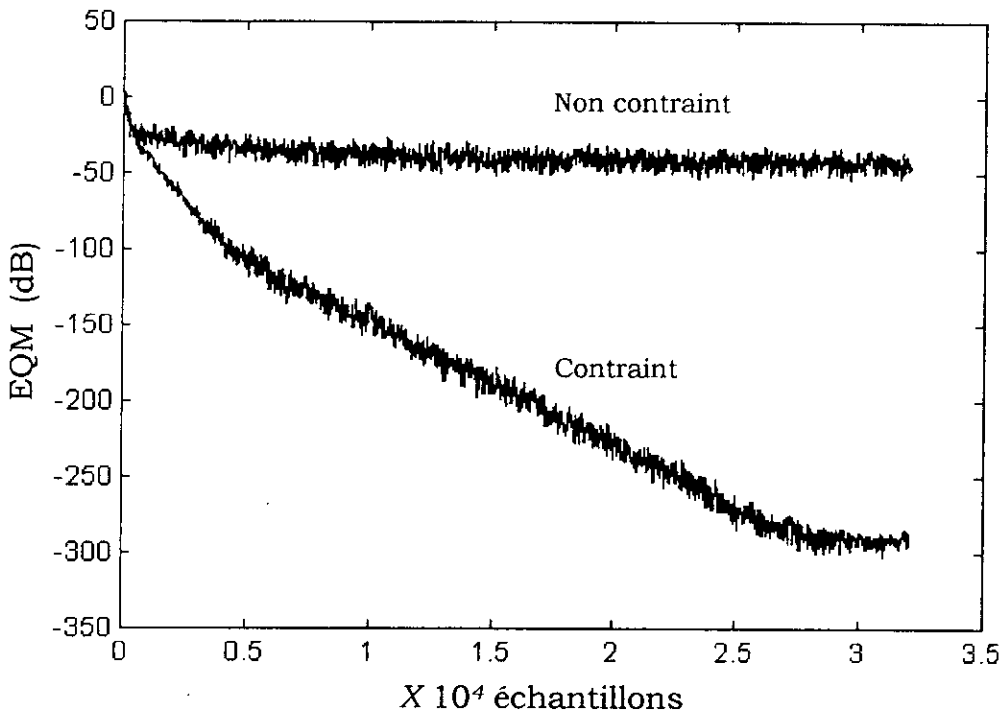

Figure 4.6 : Algorithme GMDFα contraint et non contraint pour K=4, N=16, α =1

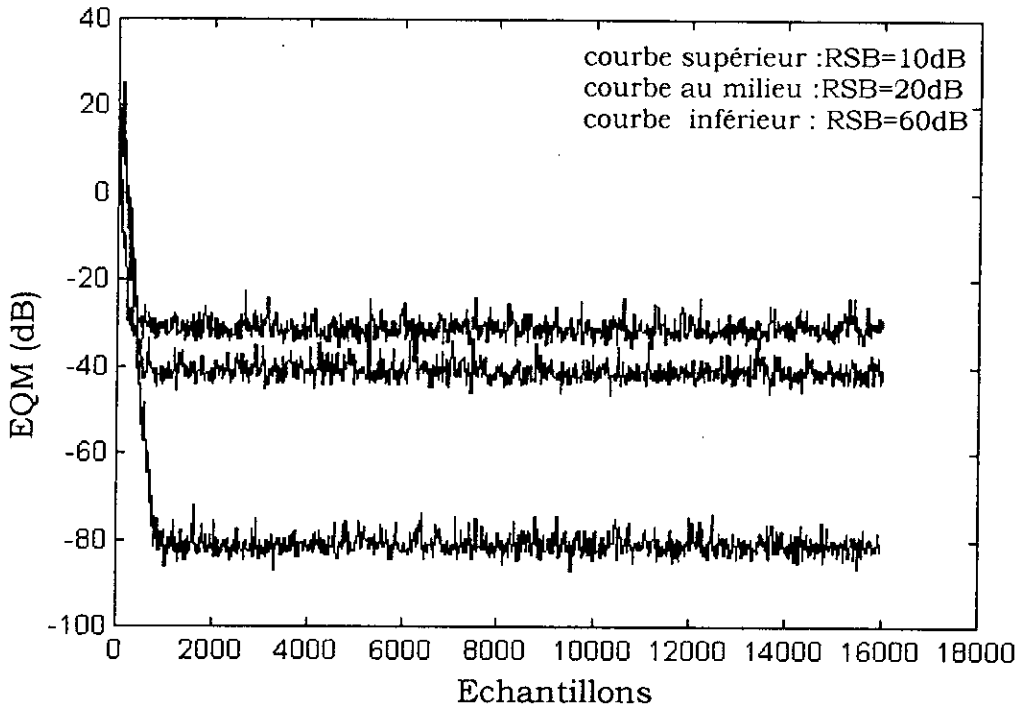

Figure 4.7 : Effet du bruit additif sur la convergence du GMDFα

4.2 Application de l'algorithme GMDFα pour l'annulation d'écho acoustique

Le but recherché par l'implémentation de l'algorithme GMDFα est son application pour l'annulation de l'écho acoustique d'un signal de parole. Nous utilisons pour cela une réponse impulsionnelle modélisant un chemin d'écho acoustique [SAN-YOU 96] dont l'expression est de la forme suivante :

$$h(i) = 0.4\ e^{-0.027i} \qquad i = 1,2,...32.$$

Nous supposons que cette réponse est fixe le long de la simulation.

La simulation (figure1.1) est basée sur l'identification d'une réponse de longueur L=32. En se référant à ce modèle, le signal $x(n)$ provenant de la salle 1 subit un écho suite au canal acoustique (la salle 2).

Dans un premier temps, nous supposons que le signal $y(n)$ est nul . De ce fait, la réponse désirée $d(n)$ (figure 4.9-b) est la convolution de $x(n)$ (figure 4.9-a) par la réponse impulsionnelle associée à la salle 2. Les figures (4.9-c et 4.9-d) illustrent l'écho résiduel $e(n)$ ainsi que l'erreur quadratique moyenne obtenus après le filtrage adaptatif.

Nous supposons maintenant, qu'au signal d'écho engendré par la salle 2 s'ajoute un bruit blanc gaussien de moyenne nulle et de variance (0.017), tel que $d(n)=x(n)*h(n) + b(n)$ (figure 4.10-b). L'écho résiduel ainsi que l'erreur quadratique moyenne sont illustrés dans les figures (4.10-c et 4.10-d).

En second lieu, un signal de parole $y(n)$ (figure 4.11-c) s'ajoute au signal d'écho $x(n)*h(n)$ tel que : $d(n)=x(n)*h(n) + y(n)$ (figure 4.11-b). L'estimée du signal de parole provenant de la deuxième salle (l'écho résiduel) ainsi que l'erreur quadratique moyenne (la différence entre le signal $y(n)$ et son estimée) sont clairement illustrés dans les figures (4.11-d et 4.11-e).

Nous supposons maintenant qu'au signal d'écho et de parole provenant de la deuxième salle s'ajoute un bruit de fond blanc gaussien de moyenne nulle et de variance (0.017), tel que : $d(n)=x(n)*h(n) + y(n) + b(n)$ (figure 4.12-b). Les figures (4.12-d et 4.12-e) illustrent de la même manière l'écho résiduel et l'erreur quadratique moyenne.

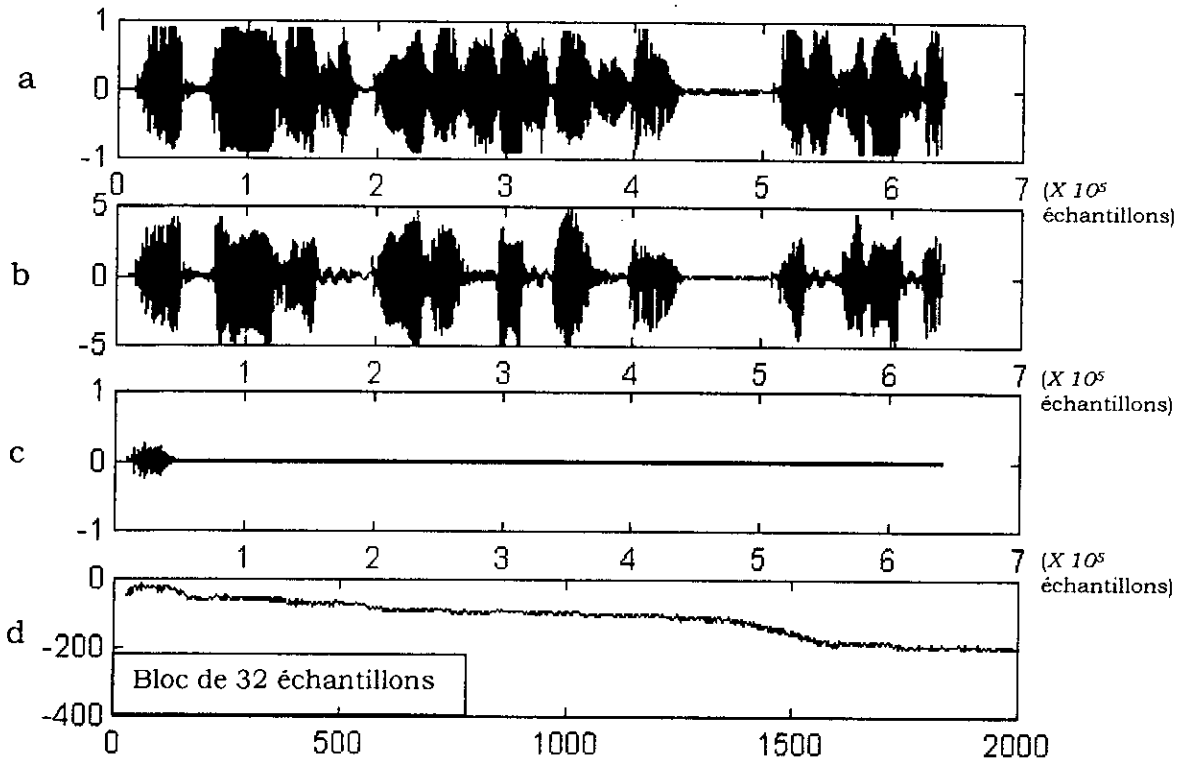

Figure 4.8 : Annulation d'écho acoustique avec $y(n)$ nul

a- Signal provenant de la salle 1

b- Signal reçu par le microphone

c- Signal d'écho résiduel

d- Erreur quadratique moyenne (dB)

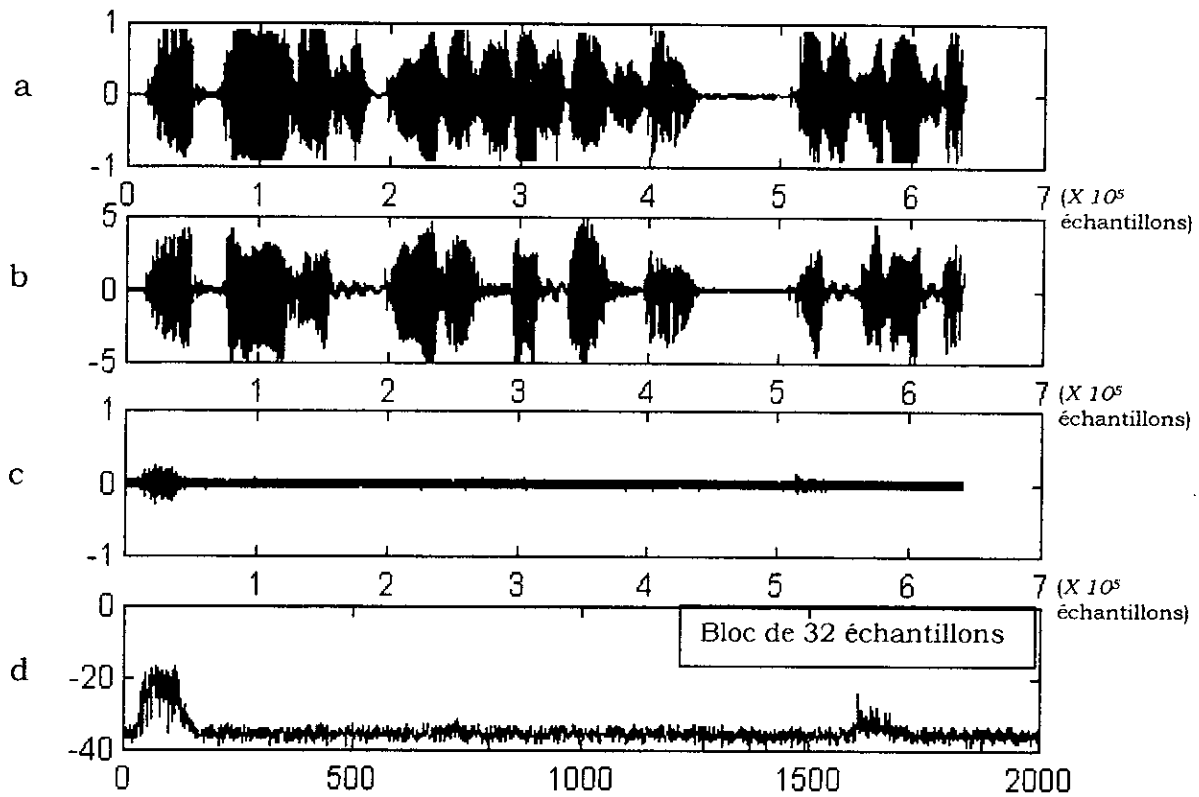

Figure 4.9 : Annulation d'écho acoustique perturbée par un bruit gaussien

a- Signal provenant de la salle 1

b- Signal reçu par le microphone

c- Signal d'écho résiduel

d- Erreur quadratique moyenne (dB)

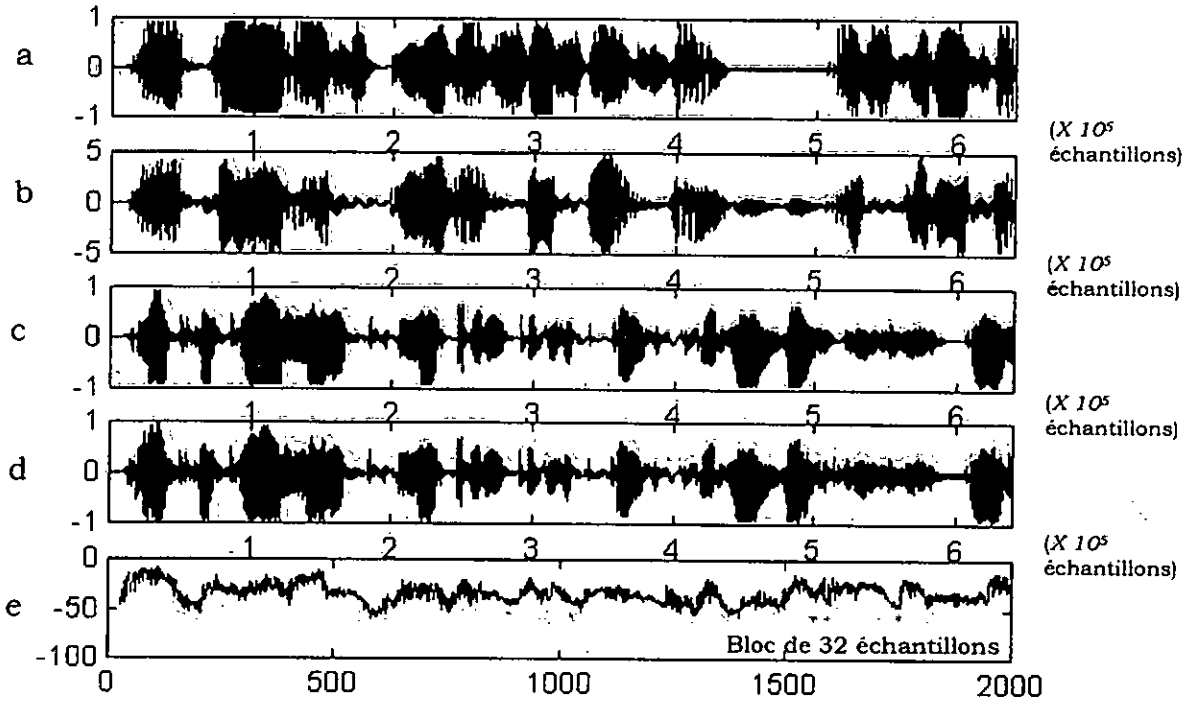

Figure 4.10 : Annulation d'écho acoustique avec existence du signal de parole $y(n)$

a- Signal provenant de la salle 1

b- Signal reçu par le microphone

c- Signal de parole $y(n)$

d- Signal transmis (d'écho résiduel)

e- Erreur quadratique moyenne (dB)

Conclusion

Notre étude s'est basée sur l'implémentation d'un annuleur d'écho acoustique en téléconférence à l'aide d'un algorithme de filtrage adaptatif dans le domaine fréquentiel.

Plusieurs algorithmes de filtrage adaptatif ont été mis au point, mais présentaient des limitations du point de vue délai de traitement, complexité arithmétique et précision. Il était donc nécessaire de concevoir un algorithme qui arrive à palier ces problèmes.

Dans ce type d'applications, où le filtrage est une tâche ardue, nous constatons que l'algorithme GMDFα est le mieux adapté, ceci au travers de :

- L'utilisation de la convolution rapide dans le domaine fréquentiel qui permet la réduction de la complexité arithmétique.

- L'utilisation du suréchantillonnage et d'une structure multi – blocs qui permettent une réduction du délai de traitement.

- L'utilisation de la normalisation du pas d'adaptation par la puissance du bloc d'entrées qui permet une convergence quasi – uniforme et plus rapide.

Nous constatons également, que la structure de l'algorithme GMDFα lui permet de couvrir une partie des algorithmes adaptatifs existant déjà :

- Pour **T=I**, $L=N$ et $\alpha=1$, l'algorithme GMDFα contraint correspond au FDLMS (Frequency Domain LMS) [GIL 92].
- Pour **T=I**, $L=N$ et $\alpha=N$, l'algorithme GMDFα contraint correspond à l'algorithme LMS dans le domaine fréquentiel avec moyennage de la sortie sur N points.
- Pour $L=N$ et $\alpha=1$, l'algorithme GMDFα contraint correspond au FD-NLMS (Frequency domain Normalized LMS) [GIL 92].

Finalement, et d'après les simulations et résultats obtenus, l'algorithme GMDFα est le mieux adapté pour les applications qui nécessitent une réponse impulsionnelle longue.

Annexe A

Nous développons dans cette annexe les équations de complexités arithmétiques correspondant aux algorithmes OLS, OLA, WOLA et à l'algorithme à délai réduit.

Les deux formules principales à utiliser [DUH 86] sont celles qui donnent le nombre d'additions et de multiplications réelles A et M respectivement pour l'algorithme de la FFT Split – Radix :

$$M_N = N/2\ (log_2N - 3\ ) + 2$$

$$A_N = N/2\ (3log_2N - 5) + 4$$

où N est la taille de la FFT.

- **Complexité de l'algorithme OLS**

Cet algorithme utilise une FFT de taille $(2N)$, une FFT inverse de la même taille et $2N$ multiplications complexes. Le nombre d'additions réelles sera donc :

$$A_{OLS} = 2\ (N(3log_2 2N - 5) + 4) + 2N = 6Nlog_2N - 2N + 8$$

celui de multiplications réelles :

$$M_{OLS} = 2\ (N(log_2N - 2\ ) + 2) + 4N = 2Nlog_2N + 4$$

- **Complexité de l'algorithme OLA**

De la même manière que pour l'algorithme OLS, cet algorithme possède le même nombre de multiplications et N additions en plus dus au recouvrement. Donc le nombre d'addition réels est :

$$A_{OLA} = 6N\log_2 N - N + 8$$

$$M_{OLS} = 2N\log_2 N + 4$$

- **Complexité de l'algorithme WOLA**

De la même manière, le nombre de multiplications reste le même pour la méthode précédante. Quant aux additions l'algorithme WOLA impose $(\alpha\text{-}1)R$ additions de plus (dus aux recouvrement également). Donc on aura :

$$A_{wola} = 6N\log_2 N\text{-}N(1+1/\alpha)+ 8$$

$$M_{OLS} = 2N\log_2 N + 4$$

- **Complexité de l'algorithme à délai réduit**

Cet algorithme utilise en plus des deux FFT, $2NK$ multiplication complexes et $2N(K-1)$ additions. On aura donc :

$$A_{DR} = 6N\log_2 N + 8 + 2N(2K-3)$$

et

$$M_{DR} = 2N\log_2 N + 4 + 4N(K-1)$$

Annexe B

Dans cette annexe, nous présentons les différents programmes de convolution rapide dans le domaine fréquentiel et de filtrage adaptatif, utilisés pour l'implémentation de l'annuleur d'écho acoustique qui fait l'objet de notre rapport.

Programme OLS

```
%Fast convolution 'OLS' algorithm
%================================
%Inputs
%======
r=input('donner le nombre de bolc :');
nb=input('donner le nombre d echantillion par blocs une puissance
de 2) ');
%set signal & impulsional response
%=================================
hp=randn(1,nb);
h=[hp zeros(1,nb)];
s=randn(1,r*nb);
x0=zeros(1,nb);
%Algorith
%========
for  k=1:r
   x=[ x0(1:nb) s((k-1)*nb+1:k*nb)];
   tx=fft(x,2*nb);
   th=fft(h,2*nb);
   ty=tx.*th;
   y1=real(ifft(ty,2*nb));
   y((k-1)*nb+1:k*nb)=y1(nb+1:nb*2);
```

```
    x0=x(nb+1:2*nb);
end
%discret convolution
%==================
c=conv(s,hp);
cd=c(1:nb*r);
%error
%=====
e=y(1:r*nb)-cd;
%graphical output
%================
i=1:r*nb;
plot(i,e);
xlabel('samples');
ylabel('error');
grid on;
```

Programme OLA

```
%Fast convolution 'ola' algorithhm
%================================
%Inputs
%======
r=input('donner le nombre de bolc :');
nb=input('donner le nombre d echantillion par blocs une puissance
de 2) ');
%set signal & impulsional response
%================================
hp=rand(1,n);
h=[hp zeros(1,nb)];
s=rand(1,r*nb);
y0=zeros(1,nb);
%Algorithm
%=========
for  k=1:r
   x=[zeros(1,nb)  s((k-1)*nb+1:k*nb)];
   tx=fft(x,2*nb);
   th=fft(h,2*nb);
   ty=tx.*th;
   y1=real(ifft(ty,2*nb));
   y((k-1)*nb+1:k*nb)=y1(nb+1:nb*2)+y0(1:nb);
   y0=y1(1:nb);
end
%discret convolution
%==================
c=conv(s,hp);
cd=c(1:nb*r);
%error
```

```
%=====
e=y(1:nb*r)-cd;
%graphical output
%================
i=1:nb*r;
plot(i,e);
xlabel('samples');
ylabel('error');
grid on;
```

Programme WOLA

```
%Fast convolution 'WOLA' algorithm
%================================
%Inputs
%======
r=input('donner le nombre de bolc :');
nb=input('donner le nombre d echantillion par blocs une puissance
de 2): ');
a=input('donner alpha: ');
d=round((nb+1)/a)-1;
%set signal & impulsional response
%================================
hp=randn(1,nb);
h=[hp zeros(1,nb)];
s1=randn(1,(r*a)*d+2*nb);
s=[ s1(1:(r*a)*d+2*nb)];
%Algorithm
%=========
for  k=1:r*a
   x=[s((k-1)*d+1:(k-1)*d+2*nb)];
   tx=fft(x,2*nb);
   th=fft(h,2*nb);
   ty=tx.*th;
   y1(k,1:2*nb)=real(ifft(ty,2*nb));
   if k>=a
      z=zeros(1,d);
      for  i=1:a
         z=z+y1(k-i+1,(nb+((i-1)*d)+1):(nb+(i*d)));
     end
       y((k-1)*d+nb+1:(k-1)*d+nb+d)=z(1:d)/a;
   end
end
%discret convolution
%===================
c=conv(s,hp);
cd=c((a-1)*d+1+nb:(((r*a)-1)*d+nb+d));
```

```
%error
%=====
e=cd-y((a-1)*d+1+nb:(((r*a)-1)*d+nb+d));
%graphical output
%================
i=1:(r-1)*a*d+d;
plot(i,e);
hold on;
xlabel('samples');
ylabel('error');
grid on;
```

Programme de l'algorithme à délai réduit

```
%Reduced delay algorithm
%Inputs
%======
r=input('donner le nombre de blocs');
nb=input(' donner le nombre d échantillons par bloc');
a=input('donner alpha');
t=input('donner K')
%set signal and impulsional response
%===================================
s=randn(1,nb*r);
h=randn(1,t*nb);
%computation of d
%================
d1=((nb+1)/a);
if ((round(d1))*a) > nb
   d=round(d1)-1;
else
   d=round(d1);
end
%Algorithm
%=========
%multi blocs structure
%=====================
km1=(((r-2)*nb+d)/d);
if ((round(km1))*d) > ((r-2)*nb+d)
   km=round(km1)-1;
else
   km=round(km1);
end
for  k=1:km
   y2=zeros(1,2*nb);
   lm=round(k*d/nb);
```

```
   if ((lm*nb) > (k*d))
      lm1=lm-1;
   else
      lm1=lm;
   end
   lm1;
    for l=1:min(lm1,t)
    x=[s((k-1)*d+1-(l-1)*nb:(k-1)*d+2*nb-(l-1)*nb)];
    tx=fft(x,2*nb);
    h1=h((l-1)*nb+1:l*nb);
    h3=[h1 zeros(1,nb)];
    th=fft(h3,2*nb);
    ty=th.*tx;
    y1=real(ifft(ty,2*nb));
    y2=y2+y1;
   end
y(k,1:2*nb)=y2(1:2*nb);
% WOLA structure
%===============
z=zeros(1,d);
      for  i=1:min(k,a)
         z=z+y(k-i+1,(nb+((i-1)*d)+1):(nb+(i*d)));
     end
      y3((k-1)*d+nb+1:(k-1)*d+nb+d)=z(1:d)/a;

end
%discret convolution
%====================
c=conv(s,h);
cd=c(1:km*d+nb);
%error
%=====
e=cd-y3(1:km*d+nb);

%graphical output
%================
i=1:km*d+nb;
plot(i,e);
xlabel('samples');
ylabel('error ');
grid on;
```

Programme du GMDFα

```
function U=gmdfu(sm,n,alfa,K,m,b,nor,cont)
%inputs are :
%sm : itterations number
% n : bloc size
% K : blocs number
% m : step size
% b : smoothing constant
% normalized algorithme if nor == 1
% constrained algorithm if cont == 1
% computation of R
%=================
r1=((n+1)/alfa);
if ((round(r1))*alfa) > n
   r=round(r1)-1;
else
   r=round(r1);
end
%impulsional response
%====================
for v=1:K
   for k=1:n
   h(v,k)=rand(1)*exp(-0.04*(k+(v-1)*n));
end
H(v,1:2*n)=fft([h(v,1:n) zeros(1,n)],2*n);
end
% initialization of the estimate of the impulsional response
%===========================================================
for c=1:K
   HE(c,1:2*n)=zeros(1,2*n);
end
%initialization of the power
%===========================
p(1:2*n)=0;
xi=zeros(1,(2*n)-r);
% GMDF Algorithme
%===============
for s=1:sm
   x1=[xi normrnd(0,2,1,r)];
   X=fft(x1,2*n);
   Xk((s-1)*r+1:(s-1)*r+2*n)=X;
   km1=round(((s-1)/alfa)+1);
   if (km1-1)*alfa > (s-1)
      km=km1-1;
```

```
else
    km=km1;
end
Yk=zeros(1,2*n);
Dk=zeros(1,2*n);
% multi bloc stucure
%==================
for i=1:min(km,K)
    XK(i,1:2*n)=Xk((s-1)*r+1-(i-1)*n:(s-1)*r+2*n-(i-1)*n);
    Yk=Yk+XK(i,1:2*n).*HE(i,1:2*n);
  Dk=Dk+XK(i,1:2*n).*H(i,1:2*n);
end
Ys=Yk;
Ds=Dk;
ds=real(ifft(Ds,2*n));
ys=real(ifft(Ys,2*n));
d=ds(n+1:n+r);
y=ys(n+1:n+r);
ys1(s,1:n)=ys(n+1);
erp(s,1:n)=ds(n+1:2*n)-ys(n+1:2*n);
% residual echo by WOLA
%======================
z=zeros(1,r);
for k=1:min(s,alfa)
    z=z+erp(s-k+1,(k-1)*r+1:k*r);
end
Eres((s-1)*n+1+n:(s-1)*n+n+r)=z/alfa;
e1=(d-y)/alfa;
% MSE
%====
for j=1:r
    u(j)=((alfa*e1(j))^2);
end
U((s-1)*r+n+1:(s-1)*r+n+r)=mean(u)/r;
%adaptaton bloc
%==============
e2=[zeros(1,n) erp(s,1:n)];
E=fft(e2,2*n);
p=p*b+(1-b)*(X.*conj(X));
for k=1:2*n
    t(k)=1/p(k);
end

if nor == 1
    T=diag(t);
else
    T=eye(2*n);
end
for i=1:min(km,K)
Xh=diag(XK(i,1:2*n));
delta=(T*Xh*E')';
```

```
    HE(i,1:2*n)=HE(i,1:2*n)+(m/2*n)*delta;
    if cont == 1
       he=(ifft(HE(i,1:2*n),2*n));
       HE(i,1:2*n)=fft([he(1:n) zeros(1,n)],2*n);
    end
    Xh=diag(zeros(1,2*n));
   end
    %clear variable
    %==============
    xi=x1(r+1:2*n);
    ds=zeros(1,n);
    e1=zeros(1,n);
    d=zeros(1,n);
    y=zeros(1,n);
    X=zeros(1,2*n);
    Ys=zeros(1,2*n);
    Ds=zeros(1,2*n);
    E=zeros(1,2*n);
    ys=zeros(1,n);
    e2=zeros(1,n);
    t=zeros(1,2*n);
    Xh=diag(zeros(1,2*n));
    delta=zeros(1,2*n);
    he=zeros(1,n);
end
%plot the MSE
%============
s=n+1:sm*r;
plot(s,10*log10(U(s)));
```

programme de l'annulation d'écho acoustique pour des signaux de parole

```
%Program of the simulation 4.12 using the GMDF algorithm
%with alfa=1,K=1.
%================
%initialisation
%==============
b=0.92;
sm=2000;
n=32;
r=32;
alfa=1;
for k=1:n
   h(k)=0.4*exp(-0.027*k);
end
HE=zeros(1,2*n);
```

```
p(1:2*n)=0.1;
m=0.005;
%set signal
%==========
xp=(wavread('signal1',400000))';
xu=(wavread('signal2',400000))';
%set impulsional response
%========================
rd=(conv(xp,h));
rd(1:sm*n+n)=rd(1:sm*n+n)+xu(1:sm*n+n);%+10^(1/2)*normrnd(0,0.13^2
,1,sm*n+n);
%principal algorithme
%====================
for s=1:sm
   %fast convolution
   %================
   x1=xp((s-1)*r+1:(s-1)*r+2*n)+noise((s-1)*r+1:(s-1)*r+2*n);
   X=fft(x1,2*n);
   Ys=X.*HE;
   ys=real(ifft(Ys,2*n));
   ds=rd((s-1)*r+1:(s-1)*r+2*n);
   %residual echo
   %==============
   e1=ds(n+1:2*n)-ys(n+1:2*n);
   Eres((s-1)*n+1+n:(s-1)*n+2*n)=e1;
   %EQM computation
   %================
   for j=1:n
      u(j)=(e1(j)-xu((s-1)*n+n+j))^2;
   end
   U(s)=10*log10((mean(u)));
   %adaptif bloc
   %============
   e2=[zeros(1,n) e1];
   E=fft(e2,2*n);
   p=p*b+(1-b)*(X.*conj(X));
   for k=1:2*n
      t(k)=1/p(k);
   end
   T=diag(t);
   Xh=diag((X));
   delta=(T*Xh*E')';
   HE=HE+(m/2*n)*delta;
   he=(ifft(HE,2*n));
   HE=fft([he(1:n) zeros(1,n)],2*n);
   %clear vaiables
   %==============
   ds=zeros(1,n);
   e1=zeros(1,n);
   X=zeros(1,2*n);
   Ys=zeros(1,2*n);
```

```
    E=zeros(1,2*n);
    ys=zeros(1,n);
    e2=zeros(1,n);
    t=zeros(1,2*n);
    Xh=diag(zeros(1,2*n));
    delta=zeros(1,2*n);
    he=zeros(1,n);
end

% plot results
%=============
s=1+n:sm*n;
subplot(5,1,1);plot(s,xp(s))
subplot(5,1,2);plot(s,rd(s))
subplot(5,1,3);plot(s,xu(s))
subplot(5,1,4);plot(s,Eres(s))
axis([17 70000 -1 1])
s=1:sm;
subplot(5,1,5);plot(s,U(s))
```

Bibliographie

[AIT 92] O.Ait Amrane, « Identification des systèmes à réponses impulsionnelle longue par filtrage adaptatif en fréquence :Application a l'annulation d'écho acoustique », Thèse de doctorat, ENST, TELECOM Paris, Octobre 1992.

[AIT-MOU-CHA-GRE 92] O.Ait Amrane, E.Moulines, M.Charbit, Y.Grenier, « LOW-DELAY FREQUENCY DOMAIN LMS ALGORITHM », Proc. IEEE ICASSP 92, pp. IV-9 IV-12.

[CHO-SES 96] K.S.K.Chow, A.B.Seasy, « Acoustic Echo Cancellation Using a Multidelay Block Hartly Domain Adaptive Filter », ICSPAT 96, Boston, pp. 83-87.

[DEN-MCC-WID 78] MAURO DENTINO, JOHN McCOOL, BERNARD WIDROW, « Adaptive Filtering in the Frequency Domain », Proc. IEEE, Vol. 66, No. 66, pp. 1658-1659, December 1978.

[DOH-POR 97] J.F.Doherty, R.Porayath, « A Robust Echo Canceler for Acoustic Environments », IEEE Trans. CAS – II, ANALOG AND DIGITAL SIGNAL PROCESSING, VOL. 44, pp. 389-396, 1997.

[DUH 86] P.Duhamel, « Implementation of ' Split-Radix' FFT Algorithms for Complex, Real, and Real-Symmetric Data », IEEE Trans. ASSP, Vol. ASSP-34, No. 2, pp. 285-295, April 1986.

[GIL 92] A.Gilloire, « Algorithmes des Moindres Carrées Récursifs Rapides à Complexité Réduite et Algorithmes par Blocs pour le Filtrage Adaptatif transverse », SSA 92, Algerie, pp.112-123.

[HAY 91] S.Haykin, « Adaptive Filter Theory », Prentice-Hall information and system sciences series, 1991.

[KIM-LEE-KIM-KIM 96] S.H.Kim, Y.H,Lee, K.D.Kim, Y.W.Kim, « Realization of a Real-Time Adaptive Acoustic Echo Canceller on ADSP-2101 », ICSPAT 96, Boston, pp. 78-82.

[LEE-MA 96] J.Lee, G.Ma, « Using Prewhitening to Iprove Performance of Multidelay Frequency Domain Adaptive Filters for Acoustic Echo Cancelletion », ICSPAT 96, Boston, pp. 34-36.

[LEE-UN 86] J.C.LEE, C.K.UN, « Performance of Trasform-Domain LMS Adaptive Digitals Filters », IEEE Trans. ASSP, Vol. ASSP-34, pp. 499-510, June 1986.

[OPP-SCH 75] A.V.Oppenheim, R.W.Schafer, « DIGITAL SIGNAL PROCESSING », Prentice- Hall, 1975.

[SOO-PAN 90] J.S.SOO, K.K.PANG, « Multydelay Block Adaptive Filter », IEEE Trans. ASSP, VOL. 38, No. 2 pp.373-376, February 1990.

[WID-STE 85] B.Widrow, S.D.Streams, « ADAPTIVE SIGNAL PROCESSING », Prentice-Hall signal processing series, 1985.