الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة الجامعات
MINISTERE AUX UNIVERSITES

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT D'ELECTRONIQUE

THESE

Présentée par LAIB Amar pour l'obtention du diplôme de

MAGISTER

Option : traitement du signal

THEME

OPTIMISATION MINIMAX DES FILTRES NUMERIQUES RIF AVEC CONTRAINTES SUPPLEMENTAIRES

Soutenue en Décembre 1992 devant le jury composé de

Mr A. CHEKIMA. . . . . .Professeur (E.N.P). . . . . Président
Mr B. DERRAS. . . . . .Ph.D (E.N.P) . . . . . Rapporteur
Mr F. CHIGARA. . . . . C. de cours (E.N.P) . . . . . Examinateur
Mr B. BOUSSEKSOU. . . . . C. de cours (E.N.P) . . . . . Examinateur
Melle M.GUERTI. . . . . C. de cours (E.N.P) . . . . . Examinateur
Mr A. GUESSOUM. . . . . Ph.D (Univ. de Blida). . . . . Invité

E.N.P 10, Avenue Hassen Badi, EL HARRACH, ALGER

الجمهوريـــة الجزائريـــة الديمقراطيـــة الشعبيـــة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة الجامعـــات

MINISTERE AUX UNIVERSITES

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT D'ELECTRONIQUE

THESE

Présentée par LAIB Amar pour l'obtention du diplôme de

MAGISTER

Option : traitement du signal

THEME

OPTIMISATION MINIMAX DES FILTRES NUMERIQUES RIF AVEC CONTRAINTES SUPPLEMENTAIRES

Soutenue en Décembre 1992 devant le jury composé de

Mr A. CHEKIMA. . . . . .Professeur (E.N.P). . . . . Président
Mr B. DERRAS. . . . . .Ph.D (E.N.P) . . . . . Rapporteur
Mr F. CHIGARA. . . . . C. de cours (E.N.P) . . . . . Examinateur
Mr B. BOUSSEKSOU. . . . . C. de cours (E.N.P) . . . . . Examinateur
Melle M.GUERTI. . . . . C. de cours (E.N.P) . . . . . .Examinateur
Mr A. GUESSOUM. . . . . Ph.D (Univ. de Blida). . . . . Invité

E.N.P 10, Avenue Hassen Badi, EL HARRACH, ALGER

DEDICACES

A mon père,
à ma mère,
à toute ma famille,
et à tous les amis

je dédie ce travail.

Amar

REMERCIEMENTS

Je tiens à remercier vivement Mr B.DERRAS pour sa large disponiblité et pour ses précieux conseils qu'il m'a prodigués tout au long de ce travail.

Je remercie également les membres du jury et du comité de lecture pour leurs précieuses suggestions. Je cite : Mr CHEKIMA, Mr BERKANI, Mr CHIGARA, Mr BOUSSEKSOU et Melle GUERTI. Qu'ils trouvent ici l'expression de ma profonde gratitude.

Mes remerciements les plus vifs vont également à Mr GORALSKI qui n'a ménagé aucun effort pour m'aider.

Enfin, je tiens à remercier tous les collègues pour leur soutien moral.

S O M M A I R E

INTODUCTION.....1

CHAPITRE I-GENERALITES SUR LES FILTRES NUMERIQUES

I-1-Introduction.....4

I-2-Généralités sur les filtres numériques

I-2-1-Définitions.....4

I-2-2-Spécifications des filtres numériques.....5

I-2-3-Classification des filtres numériques.....7

I-2-4-Comparaison entre filtres RII et filtres RIF.....8

I-3-Les filtres numériques RIF à phase linéaire.....8

I-4-Synthèse des filtres numériques RIF

I-4-1-Critères d'optimisation des filtres numériques.....10

I-4-2-Méthode d'échantillonnage en fréquence.....12

I-4-3-Méthode de fenêtrage.....12

I-5-Conclusion.....14

CHAPITRE II-PRINCIPALES METHODES D'OPTIMISATION DES FILTRES RIF A PHASE LINEAIRE SELON LA NORME DE CHEBYSHEV

II-1-Introduction.....16

II-2-Les quatre types de filtres numériques RIF à phase linéaire.....17

II-3-Méthode utilisant l'algorithme de Remez

II-3-1-Théorème de l'alternance.....19

II-3-2-Procédure de calcul de l'algorithme de Remez.....19

II-4-Méthode utilisant l'algorithme du simplexe

II-4-1-Description générale.....21

II-4-2-Formulation du problème d'optimisation minimax sous forme d'un programme linéaire.....23

II-5-Comparaison entre la méthode de Remez et la méthode du simplexe.....26

II-6-Optimisation avec contraintes supplémentaires.....27

II-7-Conclusion.....27

CHAPITRE III-UNE METHODE AMELIOREE POUR L'OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE UTILISANT L'ALGORITHME DU SIMPLEXE

III-1-Introduction.....32
III-2-Présentation de la méthode du simplexe améliorée.....32
III-3-Elaboration d'un programme et résultats
III-3-1-Description du programme.....33
III-3-2-Résultats.....39
III-3-3-Discussion du temps d'exécution.....40
III-4-Conclusion.....42

CHAPITRE IV-OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE SOUMIS A DES CONTRAINTES SIMULTANEES SUR LES CARACTERISTIQUES FREQUENTIELLE ET TEMPORELLE

IV-1-Introduction.....47
IV-2-Analyse du régime transitoire de la réponse indicielle d'un filtre numérique du type passe-bas.....47
IV-3-Limitaton des oscillations de la réponse indicielle d'un filtre numérique RIF du type passe-bas
IV-3-1-Elaboration d'un programme.....48
IV-3-2-Résultats et commentaires.....52
IV-4-Conclusion.....54

CHAPITRE V-OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE ET COEFFICIENTS DE LONGUEUR FINIE PAR UNE METHODE DE RECHERCHE LOCALE

V-1-Introduction.....58
V-2-Influence de la longueur des coefficients sur l'erreur maximale.....58
V-3-Méthode de la recherche locale
V-3-1-Position du problème.....60
V-3-2-Procédure de la recherche locale.....62
V-4-Résultats et commentaires.....64
V-5-Influence de la quantification sub-optimale sur la réponse indicielle.....66
V-6-Conclusion.....67

CONCLUSION.....72
BIBLIOGRAPHIE.....73

ANNEXES

ANNEXE I-L'algorithme du simplexe

A-Présentation théorique

A-1-Formulation d'un programme linéaire.....76

A-2-Autres définitions.....77

A-3-Résolution d'un problème de programmation linéaire.....78

B-L'algorithme du simplexe

B-1-Le tableau simplexe.....78

B-2-Organigramme de l'algorithme du simplexe.....79

B-3-Initialisation de l'algorithme du simplexe.....80

B-4-L'algorithme dual-simplexe.....80

ANNEXE II-Programmes

A-Programme FASTSIMPLEX.....87

B-Programme LOCALSEARCH.....96

INTRODUCTION

Le travail que nous présentons ici est une contribution au développement des filtres numériques RIF à phase linéaire. Il a pour objectif d'élaborer un programme pour optimiser les caractéristiques de cette classe de filtres en utilisant une méthode rapide basée sur l'algorithme du simplexe conventionnel. Cette méthode peut être considérée comme une combinaison de la procédure du simplexe et de la procédure de Remez : les deux principaux avantages de ces deux procédures y sont exploités.

* * *

La mise en évidence du filtrage numérique remonte aux années 50 [1]. En effet c'est Linville, en 1955, qui a initié, par une série de seminaires, une étude sur le filtrage numérique. A partir de cette date, la théorie de contrôle, basée essentiellement sur une uvre de Hurwicz (1945) et alimentée par une série de travaux d'éminents chercheurs, est devenue une nouvelle discipline. Ceci est dû en grande partie à la transformée en z, qui existait déjà depuis une longue date et qui se prête comme un outil mathématique descriptif des systèmes discrets.

Les applications du filtrage numérique étaient réduites, à cette époque, aux signaux qui varient lentement dans le temps [1]. Ceci est dû évidemment à la vitesse très réduite des systèmes utilisés. Et c'est vers les années 60, avec l'apparition des circuits intégrés rapides, que les filtres numériques ont connu leur essor.

* * *

Les filtres numériques, qui sont définis par leurs coefficients, peuvent être classés en deux catégories : les filtres à réponse impulsionnelle infinie (RII) ou récursifs et les filtres à réponse impulsionnelle finie (RIF) ou non récursifs.

Les filtres RII ont l'avantage de donner une réponse fréquentielle satisfaisante pour un nombre de coefficients relativement réduit en comparaison avec les filtres RIF. De plus, ils peuvent être calculés par des méthodes classiques utilisées

pour le calcul des filtres analogiques à cause des similitudes qu'ils présentent avec ces derniers [1,2].

Les filtres RIF qui présentent des avantages certains par rapport au filtres RII, telles que la stabilité et la linéarité de la phase, étaient pratiquement abandonnés à cause de la lenteur des séquences de calcul de la convolution qu'ils nécessitent surtout lorsque le nombre de coefficients est important. Et c'est grâce à l'apparition d'algorithmes rapides de calcul de la transformée de Fourier discrète (TFD), et qui ont rendu la convolution rapide possible, que les filtres RIF ont commencé à devenir le centre d'intérêt de beaucoup de chercheurs [3-7].

Les méthodes d'optimisation enregistrées dans le domaine du filtrage numérique sont nombreuses et diversifiées. Elles diffèrent, les unes des autres, par le critère d'optimisation adopté ainsi que par l'outil mathématique utilisé.

Les premiers travaux consistants réalisés dans ce domaine remontent aux années 1969-1970. En 1969, B.Gold et K.L.Jordan ont publié un article décrivant une procédure de calcul direct des filtres numériques RIF [3], suivis, en 1970, par L.R. Rabiner, B.Gold et C.A.McGonegal qui ont présenté une nouvelle méthode d'approximation des filtres numériques RIF [4]. Et ce n'est qu'en 1972 que le premier programme efficace d'optimisation des filtres numériques RIF à phase linéaire au sens de Chebyshev est apparu [5]. Présenté par Thomas W.Parks et James McClellan, ce programme très utilisé est basé sur l'algorithme rapide de Remez. Ce travail a été repris, en 1973, par les mêmes auteurs [6] en vu d'une généralisation pour les différents types de filtres numériques RIF à phase linéaire. Ce programme qui utilise l'algorithme de Remez, est considéré, et à ce jour, comme le plus rapide des programmes de calcul des filtres numériques RIF à phase linéaire au sens de la norme de Chebyshev.

En même temps, un autre programme d'optimisation des filtres numériques RIF selon la norme de Chebyshev basé sur la programmation linéaire (algorithme du simplexe) fut publié, en 1972, par L.R.Rabiner [7]. Ce dernier fut rapidement abandonné à cause de sa lenteur bien qu'il présente un avantage certain par

rapport au programme utilisant l'algorithme de Remez à savoir, la possibilité d'associer des contraintes supplémentaires aux contraintes principales (gabarit fréquentiel).

Depuis lors, la méthode rapide utilisant l'algorithme de Remez et la méthode très lente (mais plus générale) utilisant l'algorithme du simplexe continuent à constituer les principales méthodes d'optimisation des filtres numériques RIF à phase linéaire.

* * *

Notre travail est organisé comme suit. Le premier chapitre traitera des généralités sur les filtres numériques. Nous y avons introduit une étude comparative entre les filtres RII et les filtres RIF pour faire apparaître l'importance de ces derniers.

Le second chapitre est une description détaillée des deux principales méthodes utilisées pour l'optimisation des filtres RIF selon la norme de Chebyshev, à savoir : la méthode utilisant l'algorithme de Remez et la méthode utilisant l'algorithme du simplexe .

Dans le troisième chapitre, nous développerons une technique rapide pour l'opimisation des filtres RIF à phase linéaire. Elle se présente comme une amélioration de l'ancienne méthode utilisant l'algorithme du simplexe.

Le quatrième chapitre, qui se présente comme la suite du chapitre précédent, traitera de l'optimisation avec contraintes supplémentaires et nous l'avons organisé à part pour faire apparaître l'importance de cet avantage propre à la méthode utilisant l'algorithme du simplexe.

Enfin, le dernier chapitre sera consacré à l'optimisation des filtres RIF à coefficients de longueur finie (ou quantifiés) par une méthode de recherche locale.

* * *

CHAPITRE I

GENERALITES SUR LES FILTRES NUMERIQUES

I-1-INTRODUCTION

Le but visé dans ce chapitre est de donner une idée générale sur les filtres numériques. Il représente une base théorique indispensable pour notre travail. Une priorité y est donnée aux filtres numériques à réponse impulsionnelle finie : il s'agit d'une classe spécifique de filtres numériques auxquels on accorde aujourd'hui une importance capitale.

Le présent chapitre est organisé comme suit. Après un bref rappel sur les définitions couramment employées dans le thème du filtrage numérique, ainsi que les spécifications des filtres numériques, nous donnerons une classification de ces derniers selon la durée de leur réponse impulsionnelle et qui sera suivie d'une étude comparative pour montrer l'importance des filtres numériques à réponse impulsionnelle finie. Ces derniers bénéficient, dans ce chapitre, d'une étude relativement détaillée.

I-2-GENERALITES SUR LES FILTRES NUMERIQUES

I-2-1-Définitions

D'une manière générale, le *filtrage numérique* est toute opération en traitement du signal dans laquelle le signal numérique d'entrée x et le signal numérique de sortie y sont liés par une relation du type:

$$y(nT) = \sum_{i=0}^{m} a_i x((n-i)T) - \sum_{j=1}^{n} b_j y((n-j)T) \qquad (I\text{-}1)$$

où les coefficients a_i et b_j sont des constantes et T représente la période d'échantillonnage dans le domaine temporel (normalisée souvent à l'unité).

Les *filtres numériques* peuvent être définis comme étant les systèmes numériques qui assurent cette fonction. Ils sont donc des systèmes linéaires invariants dans le temps.

L'équation aux différences (I-1) caractérise entièrement le filtre numérique. La transformée en z appliquée à cette équation nous permet d'écrire [1]:

$$Y(z) = H(z).X(z) \qquad (I-2)$$

avec

$$H(z) = \frac{\sum_{i=0}^{m} a_i z^{-i}}{1 + \sum_{j=1}^{n} b_j z^{-j}} \qquad (I-3)$$

H(z) est appelée *fonction de transfert* et la *réponse fréquentielle* est obtenue en remplaçant la variable z par le terme $e^{j2\pi f}$.

Soit h(i) la transformée inverse en z de H(z), on a:

$$H(z) = \sum_{i=-(m-n)}^{+\infty} h(i).z^{-i} \qquad (I-4)$$

$$h(i) = \frac{1}{2\pi j} \oint_{|z|=1} H(z).z^{i-1}.dz \qquad (I-5)$$

et comme:

$$z = e^{j2\pi f} \qquad (I-6)$$

il vient:

$$h(i) = \int_{-\infty}^{+\infty} H(e^{j2\pi f}).e^{j2\pi if}.df \qquad (I-7)$$

h(i) est appelée *réponse impulsionnelle* du filtre numérique.

I-2-2-Spécifications des filtres numériques

Les filtres numériques peuvent donc être spécifiés aussi bien par leur réponse temporelle h(i) que par leur réponse fréquentielle $H(e^{j2\pi f})$. Les spécifications temporelles sont, en général, réservées au filtrage adaptatif [2]. Par contre, les spécifications fréquentielles couvrent la plupart des autres types de filtrage.

Dans ce dernier cas, l'objectif visé est de déterminer les coefficients du filtre numérique dont la réponse fréquentielle approche au mieux une réponse désirée tout en restant inscrite dans un gabarit. Deux exemples de gabarit sont illustrés dans les figures I-1 et I-2, les fréquences sont normalisées à la fréquence d'échantillonnage qui vaut l'unité.

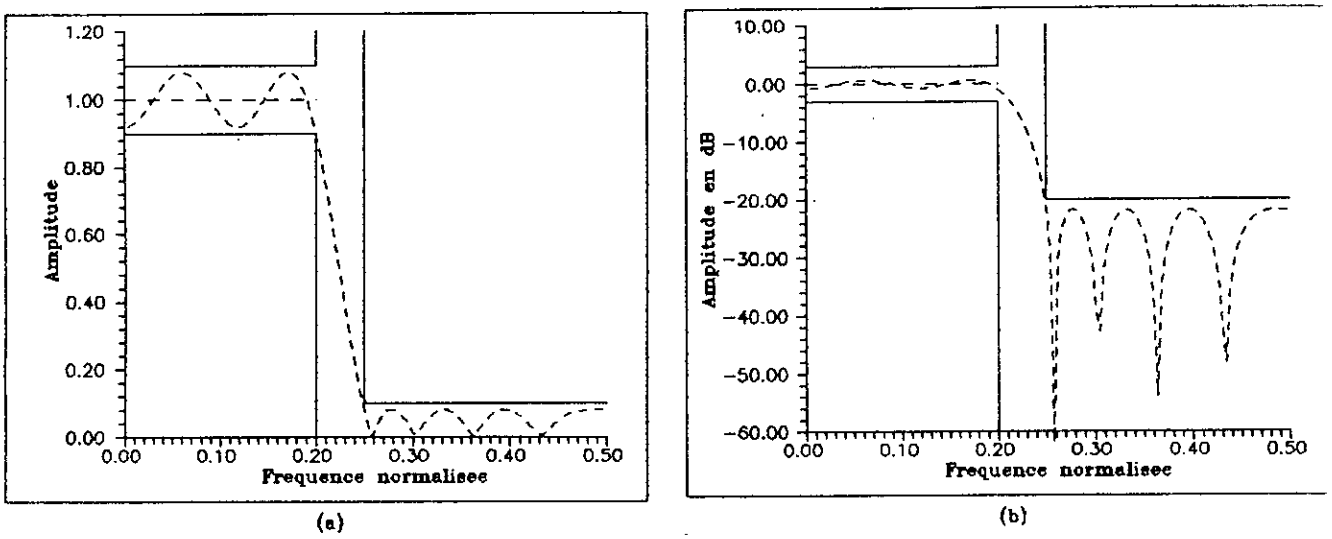


Fig.I-1-Gabarit frequentiel d'un filtre passe-bas
a-Representation en echelle lineaire.
b-Representation en decibels (dB)

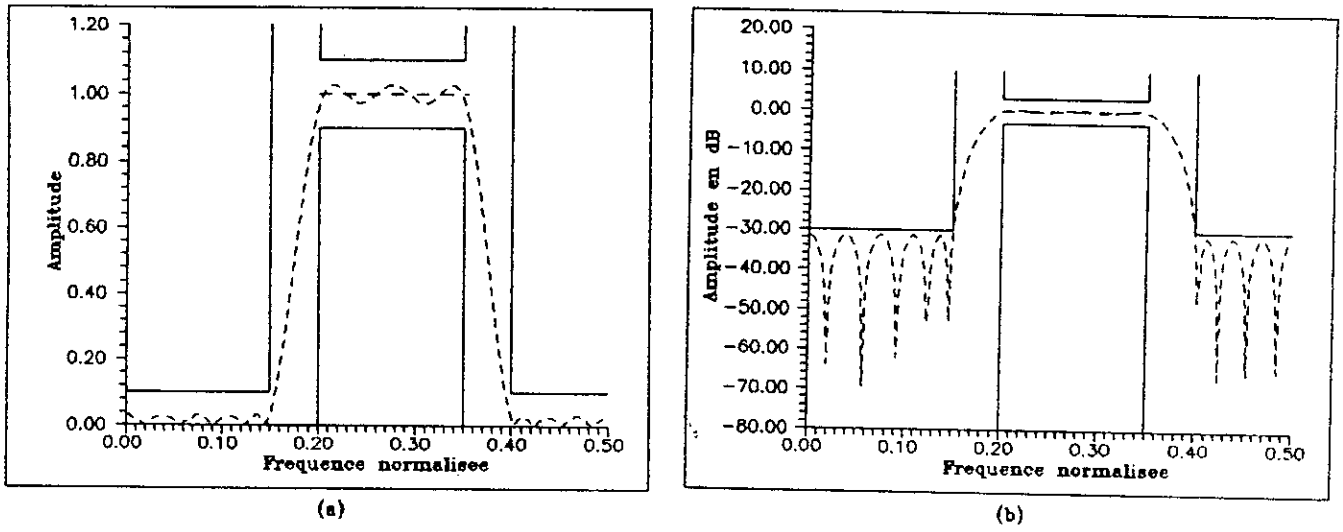


Fig.I-2-Gabarit frequentiel d'un filtre passe-bande.
(a)-Representation en echelle lineaire.
(b)-Representation en decibels (dB).

Une autre représentation du gabarit est la suivante. Nous pouvons définir le gabarit par deux fonctions : une fonction à approcher ou fonction objectif D(f), et une fonction de pondération W(f). Cette dernière exprime le rapport de l'écart maximum dans chaque bande sur l'écart maximum dans une certaine bande pris comme quantité à minimiser.

Ainsi, les gabarits des figures I-1 et I-2 sont couramment représentés, respectivement, par les expressions suivantes:

$$D(f)=\begin{cases} 1 \text{ pour } f \in [0.00,0.20] \\ 0 \text{ pour } f \in [0.25,0.50] \end{cases}, \qquad W(f)=\begin{cases} 1 \text{ pour } f \in [0.00,0.20] \\ 1 \text{ pour } f \in [0.25,0.50] \end{cases}$$

pour le filtre-passe bas et

$$D(f)=\begin{cases} 0 \text{ pour } f \in [0.00,0.15] \\ 1 \text{ pour } f \in [0.20,0.35] \\ 0 \text{ pour } f \in [0.40,0.50] \end{cases}, \qquad W(f)=\begin{cases} 1 \text{ pour } f \in [0.00,0.15] \\ 1 \text{ pour } f \in [0.20,0.35] \\ 1 \text{ pour } f \in [0.40,0.50] \end{cases}$$

pour le filtre passe-bande.

Ici la pondération est la même pour toutes les bandes. Si on veut renforcer l'atténuation dans une certaine bande, il suffit de choisir une pondération plus élevée dans celle-ci.

I-2-3-Classification des filtres numériques

Selon leur réponse temporelle, les filtres numériques sont classés en deux catégories:

-*Les filtres numériques à réponse impulsionnelle finie* (RIF), dont la réponse impulsionnelle présente un nombre fini de coefficients non nuls. Tous les autres coefficients sont nuls.

Pour de tels filtres, la fonction de transfert est donnée par:

$$H(e^{j2\pi f}) = \sum_{i=-N1}^{N2} h(i).e^{-j2\pi if} \qquad (I-8)$$

-*Les filtres numériques à réponse impulsionnelle infinie* (RII), dont la réponse impulsionnelle présente un nombre infinie d'échantillons non nuls.

Pour ce type de filtres, la fonction de transfert est donnée par:

$$H(e^{j2\pi f}) = \sum_{i=-(m-n)}^{+\infty} h(i).e^{-j2\pi i f} \qquad (I-9)$$

I-2-4-Comparaison entre filtres RIF et filtres RII

L'inconvénient majeur des filtres numériques RIF, est l'importance du nombre de coefficients qu'ils nécessitent pour avoir une réponse fréquentielle satisfaisante. Alors que, pour avoir la même réponse fréquentielle, les filtres RII ne nécessiteraient qu'un nombre relativement faible .

En contre-partie, les filtres RIF répondent à des formulations mathématiques très simples et sont, par conséquent, faciles à implémenter. D'autre part, leur stabilité est toujours assurée, ce qui n'est pas le cas pour les filtres numériques RII. De plus, ils peuvent donner une phase exactement linéaire et l'absence de contre-réaction de la sortie sur l'entrée fait que ces derniers soient moins sensibles aux erreurs de calcul.

I-3-LES FILTRES NUMERIQUES A REPONSE IMPULSIONNELLE FINIE A PHASE LINEAIRE

Causalité et stabilité

Pour qu'un filtre numérique RIF (ou RII) soit techniquement réalisable, il faut qu'il soit causal et stable, c'est-à-dire, il faut que sa réponse impulsionnelle vérifie les deux relations suivantes [1,8]:

$$h(i)=0 \quad , \qquad \forall\ i < 0 \tag{I-10}$$

et

$$\lim_{i \to \infty} h(i)=0 \tag{I-11}$$

Pour un filtre numérique RIF, nous pouvons écrire:

$$H(z)=\sum_{i=0}^{N-1} h(i).z^{-i} \tag{I-12}$$

d'où

$$H(f)=\sum_{i=0}^{N-1} h(i).e^{-j2\pi if} \tag{I-13}$$

L'entier N, qui représente le nombre de coefficients, est aussi appelé *ordre du filtre*.

Linéarité de la phase

Un filtre est dit à phase linéaire, si la phase de sa réponse fréquentielle ϕ vérifie la relation suivante:

$$\phi(f)=2\pi af + b \qquad f \in \mathcal{F} \qquad (I-14)$$

où a et b sont des constantes réelles et $\mathcal{F}$ représente l'ensemble des bandes où le filtre est défini.

Pour que la phase d'un filtre numérique RIF soit linéaire, il faut et il suffit que sa réponse impulsionnelle soit symétrique ou anti-symétrique [1]. C'est-à-dire, il faut que ses coefficients vérifient l'une des deux relations suivantes:

$$h(i)=h(N-i-1), \text{ pour } i=0,\ldots,N-1 \qquad (I-15-a)$$

ou bien,

$$h(i)=-h(N-i-1), \text{ pour } i=0,\ldots,N-1 \qquad (I-15-b)$$

La relation (I-15-a) signifie que la réponse impulsionnelle est symétrique (ou possède une symétrie positive), et la relation (I-15-b) signifie que celle-ci est anti-symétrique (ou possède une symétrie négative).

Supposons que l'une de ces deux relations soit vérifiée et qu'en plus, N est impair (N=2n+1). La relation (I-13) peut être réécrite sous la forme:

$$H(f) = h(0)+h(1).e^{-j2\pi f} + \ldots + h(n-1).e^{-j2\pi(n-1)f} + h(n).e^{-j2\pi nf}$$
$$+ h(n+1).e^{-j2\pi(n+1)f} + \ldots + h(2n-1).e^{-j2\pi(2n-1)f} + h(2n).e^{-j4\pi nf}$$

avec

$$n = \frac{N-1}{2}$$

En utilisant la relation (I-15-a) ou (I-15-b), nous pouvons écrire:

$$H(f) = h(0).(1 \pm e^{-j4\pi nf}) + h(1).(e^{-j2\pi f} \pm e^{-j2\pi(2n-1)f}) + \ldots$$
$$+ h(n-1).(e^{-j2\pi(n-1)f} \pm e^{-j2\pi(n+1)f}) + h(n).e^{-j2\pi nf}.$$

ou bien

$$H(f)=\left[h(0).(e^{j2\pi nf} \pm e^{-j2\pi nf}) + h(1).(e^{j2\pi(n-1)f} \pm e^{-j2\pi(n-1)f}) + \right.$$
$$\left. \ldots + h(n-1).(e^{j2\pi f} \pm e^{-j2\pi f}) + h(n)\right].e^{-j2\pi nf}$$

et en utilisant les deux formules suivantes:

$$e^{jx} + e^{-jx} = 2.\cos(x)$$
$$e^{jx} - e^{-jx} = 2.\cos(x).e^{j\frac{\pi}{2}}$$

il vient

$$H(f)=G(f).\exp\left[j\left(\frac{L\pi}{2}-\frac{(N-1)}{2}2\pi f\right)\right] \qquad (I-16)$$

où G(f) est une fonction réelle et

$$L=\begin{cases} 0 & \text{pour une réponse impulsionnelle symétrique.} \\ 1 & \text{pour une réponse impulsionnelle anti-symétrique.} \end{cases}$$

La relation (I-16) exprime bien que la phase est linéaire.

Si N est pair, le même raisonnement nous conduit à la même conclusion.

En toute rigueur, la relation (I-16) n'implique pas la linéarité de la phase car les changements de signe de la fonction réelle G(f) entraînent des discontinuités de π sur la phase. Cependant, par extension, les filtres vérifiant la condition suscitée sont dits à phase linéaire [2].

Nous verrons plus loin (cf. § II-2) qu'il est possible d'envisager quatre cas de filtres numériques à phase linéaire et ce, selon que le nombre de coefficients est pair ou impair et selon que la réponse impulsionnelle est symétrique ou anti-symétrique.

I-4-SYNTHESE DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE

La synthèse des filtres numériques passe par la détermination des coefficients qui satisfont à des contraintes données comme première étape, ensuite, par le choix d'une structure de réalisation comme seconde étape.

Pour la première étape, celle qui nous intéresse, plusieurs méthodes [1,2,8-10] ont été déjà décrites dans la littérature. Il n'éxiste pas une méthode générale qui s'applique à tous les cas car les contraintes ne sont pas les mêmes pour tous les types de réalisations. Néanmoins, si on ne considère que les contraintes principales (celles qui sont liées uniquement au gabarit fréquentiel), nous pouvons classer ces méthodes en un nombre fini de catégories.

I-4-1-Critères d'optimisation des filtres numériques

La détermination des coefficients des filtres numériques repose sur des critères d'optimisation. Ces derniers peuvent être classés en deux catégories:

-*Les critères principaux* qui concernent le gabarit fréquentiel et qui sont traduits par des contraintes sur l'écart entre la réponse fréquentielle du filtre calculé et la réponse fréquentielle désirée.

-*Les critères additifs* qui sont, en général, des contraintes supplémentaires qui concernent la représentation temporelle du filtre. Nous pouvons citer l'exemple de l'optimisation de la réponse indicielle d'un filtre numérique RIF à phase linéaire que nous développerons plus loin (Chap.IV).

L'écart (ou erreur) entre la réponse fréquentielle du filtre calculé (H(f)) et la réponse fréquentielle désirée ((D(f)) peut être formulé comme suit:

$$E(f) = W(f).[H(f) - D(f)] \qquad \text{(I-17)}$$

où W(f) est une fonction de pondération qui permet de contrôler l'erreur dans le gabarit. Elle doit vérifier la relation suivante:

$$W(f) > 0 \quad \forall\ f \in \mathcal{F} \subseteq [0.,0.5] \qquad \text{(I-18)}$$

$\mathcal{F}$ étant la réunion de tous les intervalles où D(f) est définie (bandes passantes et coupées).

Le problème d'optimisation revient à déterminer la réponse fréquentielle H(f) du filtre qui minimise une fonction erreur E(f) au sens de la norme L_p. Soit:

$$\| E \|_{Lp} = \left(\int_{-1/2}^{1/2} \| E(f) \|^p . df \right)^{\frac{1}{p}} \qquad \text{(I-19)}$$

avec

$$p \in \mathbb{N}^* \quad \text{et} \quad f \in \mathcal{F}$$

Dans le cas d'un traitement numérique, le problème précédent nécessite une discrétisation. Toutes les grandeurs fréquentielles sont alors échantillonnées et la norme L_p de l'erreur est définie, dans ce cas, comme suit:

$$\| E \|_{Lp} = \left(\sum_{k=0}^{Ne-1} \| E(f_k) \|^p \right)^{\frac{1}{p}} , \quad f_k \in \mathcal{F} \qquad \text{(I-20)}$$

Le pas d'échantillonnage doit être suffisamment petit pour que l'information contenue dans le gabarit soit intégralement prise en considération. Le nombre d'échantillons Ne est pris, en général, proportionnel à l'ordre du filtre. Le facteur de proportionnalité,

qui exprime la densité de grille, est un facteur très important dans le volume de calcul du filtre [11].

I-4-2-Méthode d'échantillonnage en fréquence

Cette méthode [1,2] utilise la transformée de Fourier discrète (T.F.D) pour laquelle des algorithmes de calcul très rapides ont été déjà élaborés; la réponse désirée est échantillonnée en N points (N étant l'ordre du filtre). Ensuite, les N échantillons de la réponse impulsionnelle sont obtenus par le calcul de la T.F.D inverse, soit

$$h(i) = \frac{1}{N} \sum_{k=0}^{N-1} D(k).\exp(j2\pi ik/N) \qquad (I-21)$$

Les coefficients ainsi calculés ne correspondent pas forcément à la solution optimale. La réponse fréquentielle du filtre calculée coïncide avec la réponse désirée mais seulement en les N points donnés par l'échantillonnage. Pour le reste des points, l'erreur reste incontrôlée.

L'optimisation se fait en ajustant les coefficients de telle manière à minimiser la norme L_p de l'erreur donnée par l'expression (I-20) (pour une valeur donnée de p).

I-4-3-Méthode de fenêtrage

La réponse fréquentielle désirée est supposée périodique, on calcule les coefficients conformément à la transformation de Fourier inverse [1]:

$$h(i)=\int_{-1/2}^{1/2} D(f).e^{j2\pi if}df \qquad (I-22)$$

Le nombre de coefficients, infini au départ, est limité par une fenêtre wd(n) de largeur N (qu'on supposera pair). Les coefficients du filtre sont alors:

$$h_D(i) = h(i).wd(i) \qquad (I-23)$$

avec

$$wd(i)=0 \quad \text{pour } i < -N/2 \text{ et } i > N/2 \qquad (I-24)$$

Dans le domaine fréquentiel ceci est équivalent à une convolution de la réponse fréquentielle désirée avec la transformée de Fourier de la fenêtre (soit WD(f)). La réponse fréquentielle du

filtre calculée par cette méthode est le résultat de cette convolution:

$$H_D(f) = D(f)*WD(f) \qquad (I-25)$$

où $H_D(f)$ est la transformée de Fourier directe de la réponse impulsionnelle $h_D(k)$.

Il en résulte des ondulations indésirables (phénomène de Gibbs) qu'il faut minimiser en choisissant un type de fenêtre convenable. En réalité, ces ondulations ne sont pas les seuls paramètres à optimiser, il y a aussi la largeur de la bande de transition. Pour plus de détails nous préférons orienter le lecteur vers la référence [1]. Dans celle-ci, il trouvera une étude plus abondante concernant la synthèse des filtres numériques RIF par la méthode de fenêtrage.

Fenêtre rectangulaire

C'est une simple troncature, elle est définie par:

$$wd(n) = \begin{cases} 1 & \text{pour } -\frac{N}{2} \le n \le \frac{N}{2} \\ 0 & \text{ailleurs} \end{cases}$$

La figure I-3 illustre un exemple de réponse fréquentielle d'un filtre passe-bas défini par le gabarit de la figure I-1 et calculé par la méthode de fenêtrage en utilisant une fenêtre rectangulaire pour des largeurs différentes. Nous pouvons constater que les ondulations sont d'autant plus faibles que la fenêtre est large (Fig.I-3). Les coefficients du filtre avant l'application du fenêtrage sont obtenus par la relation (I-22).

Fenêtre de Hamming généralisée

Elle est définie par:

$$wd(n) = \begin{cases} \alpha + (1-\alpha).\cos(2\pi n/N) & \text{pour } -\frac{N}{2} \le n \le \frac{N}{2} \\ 0 & \text{ailleurs} \end{cases}$$

La figure I-4 représente la réponse fréquentielle du filtre numérique défini précédemment calculé en utilisant la fenêtre de Hamming ($\alpha = 0.54$) pour différentes largeurs.

Comparaison

Pour une même largeur, nous pouvons constater que la fenêtre de Hamming entraîne des ondulations beaucoup plus amortis en comparaison avec les ondulations qu'entraîne la fenêtre rectangulaire. Mais en examinant attentivement la courbe de la figure I-5 nous pouvons constater qu'en contre-partie la fenêtre de Hamming élargit la bande de transition et cela à cause du lobe principal de celle-ci qui est le double de celui de la fenêtre rectangulaire.

I-5-CONCLUSION

Nous n'avons retenu, dans ce chapitre, que les points essentiels sur les filtres numériques RIF à phase linéaire. Lesquels points sont nécessaires pour le développement des chapitres suivants. Pour plus de détails et d'approfondissements, le lecteur trouvera dans la bibliographie une selection de références qui offrent une étude beaucoup plus abondante. Nous lui conseillons les références [1,2,8] et plus particulièrement l'ouvrage [1].

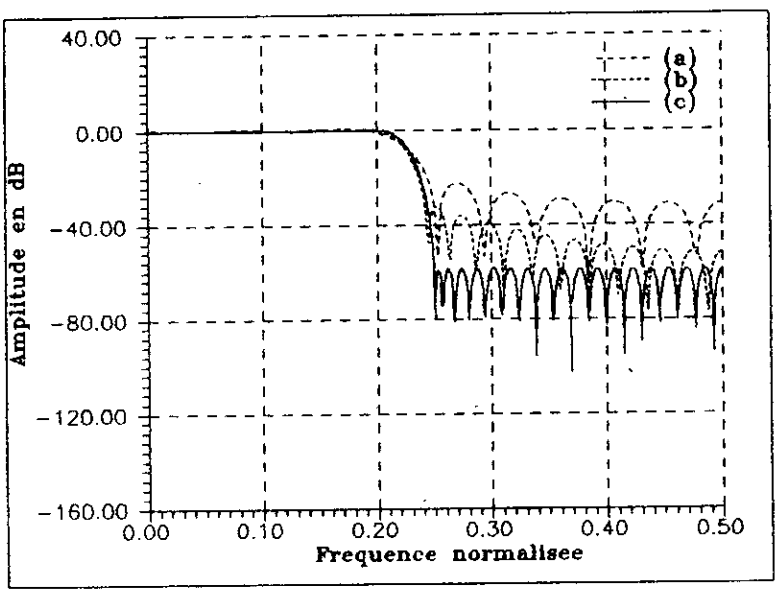


Fig.I-3-Spectre d'amplitude d'un filtre numerique passe-bas (defini par le gabarit de la figure I-1) limite par une fenetre rectangulaire de largeur:
(a) 21 (b) 41 (c) 65

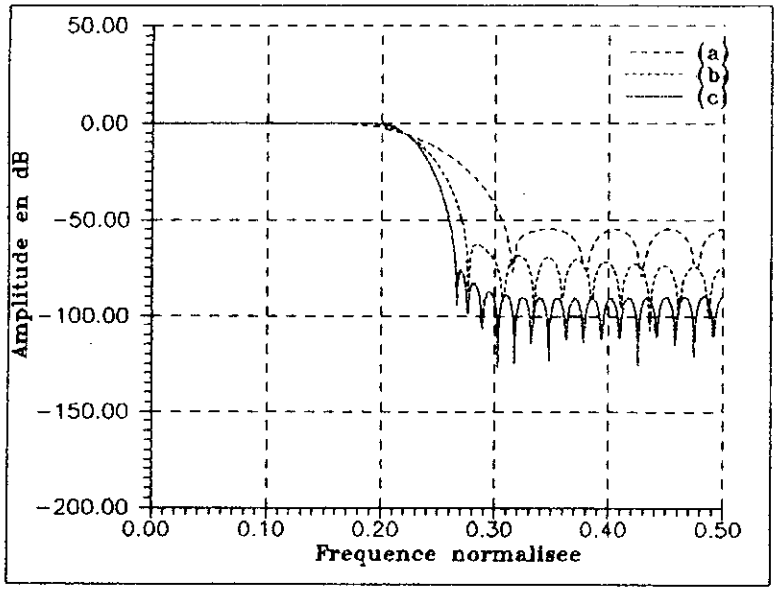


Fig.I-4-Spectre d'amplitude d'un filtre numerique passe-pas (defini par le gabarit de la figure I-1) limite par une fenetre de Hamming de largeur:
(a) 21 (b) 41 (c) 65

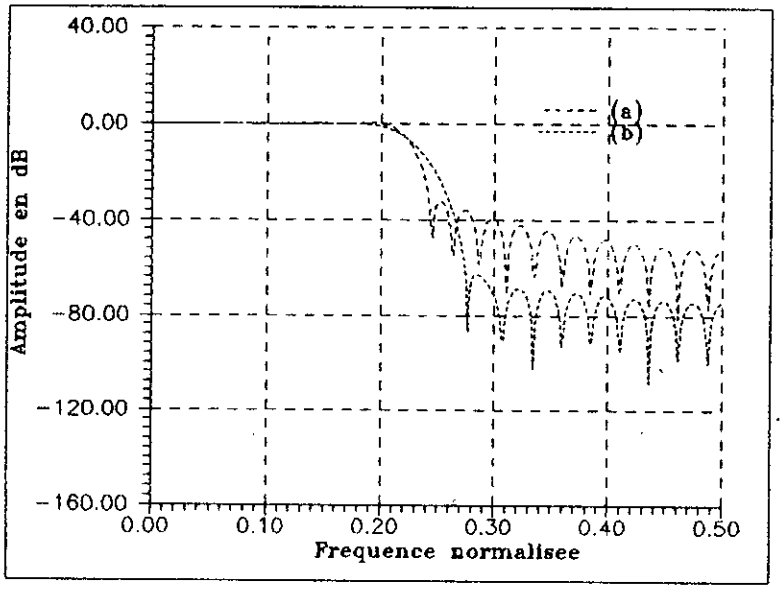


Fig.I-5-Spectre d'amplitude d'un filtre numerique passe-bas (defini par le gabarit de la figure I-1), limite par deux types de fenetres de meme largeur (egale a 41):
(a)-Fenetre rectangulaire.
(b)-Fenetre de Hamming.

CHAPITRE II

PRINCIPALES METHODES D'OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE SELON LA NORME DE CHEBYSHEV

II-1-INTRODUCTION

Le présent chapitre offre une description sur les deux principales méthodes utilisées dans le calcul des filtres numériques RIF pour des spécifications fréquentielles répondant au critère de Chebyshev à savoir : la méthode de Remez et la méthode du simplexe. Ces deux méthodes ont l'avantage de minimiser le maximum de l'erreur (donc de contrôler l'erreur en tout point de l'axe fréquentiel). Elles s'opposent aux autres méthodes, telles que la méthode des moindres carrés ou la méthode des valeurs absolues, qui se limitent à minimiser une certaine norme de l'erreur sans se soucier des variations de celle-ci sur l'axe fréquentiel.

Les deux méthodes que nous allons développer reposent sur le critère de la norme L_∞. Le problème d'optimisation revient, dans ce cas, à déterminer la réponse fréquentielle du filtre qui minimise le maximum de l'erreur où elle est définie. On dit aussi qu'on a affaire à un problème de minimax [11].

On sait que la réponse fréquentielle H(f) d'un filtre numérique est définie par ses coefficients. Donc, déterminer la réponse fréquentielle d'un filtre numérique d'ordre N, revient à calculer ses coefficients h(i) (i=0,...,N-1). Le problème à traiter peut être formulé comme suit:

$$\underset{(h(i),\,i=0,\ldots,n)}{\text{minimiser}} \left(\max_{f\in\mathcal{F}} \left(W(f).\|H(f)-D(f)\| \right) \right) \qquad \text{(II-1)}$$

La fonction approximante H(f) est une combinaison linéaire de N fonctions formant une base orthogonale, soit:

$$H(f)=\sum_{i=0}^{N-1} h(i)\exp(-j2\pi i f) \qquad \text{(II-2)}$$

Si la phase est linéaire, et si N est impair (N=2n+1), nous

pouvons écrire alors (c.f § I-3):

$$H(f)=G(f).\exp(j(L\frac{\pi}{2}-2\pi nf)) \qquad \text{(II-3)}$$

où G(f) est une fonction réelle.

Les méthodes utilisées pour traiter ce problème sont de deux sortes : celles qui reposent sur l'algorithme de Remez et celles qui utilisent la programmation linéaire (algorithme du simplexe).

II-2-LES QUATRE TYPES DE FILTRES RIF A PHASE LINEAIRE

Selon que N est pair ou impair et selon que la réponse impulsionnelle est symétrique ou anti-symétrique, on peut envisager quatre types de filtres RIF et à phase linéaire [**1,11,6**] qui peuvent être résumés par le tableau II-1.

Les expressions de G(f) ne sont pas les mêmes pour les différents cas. Mais, pour le calcul du filtre numérique on peut prendre une forme générale unique du type

$$G(f)=\sum_{i=0}^{n} d_i.\cos(2\pi if) \qquad \text{(II-4)}$$

L'optimisation s'effectue alors en traitant un problème équivalent [**11,6**] dans lequel la fonction objectif D(f) et la fonction de pondération W(f) prennent de nouvelles expressions. Les coefficients h(i) du filtre sont ensuite obtenus directement à partir des coefficients calculés d_i solutions du problème équivalent.

Le problème équivalent peut être formulé comme suit:

$$\underset{(d_i,\,i=0,\ldots,n)}{\text{minimiser}} \left(\max_{f\in\mathcal{F}} \left(\hat{W}(f).|G(f) - \hat{D}(f)| \right) \right) \qquad \text{(II-5)}$$

où $\hat{D}(f)$ et $\hat{W}(f)$ sont les nouvelles expressions de D(f) et W(f) pour lesquelles il faut opter au départ et qui sont résumées, pour les quatre cas, dans le tableau II-2. On y trouve également les relations qui permettent de retrouver les coefficients du filtre h(i) à partir des coefficients calculés d_i.

Tableau II-1-Expressions de G(f) dans les quatre cas.

Symétrie → / ↓ Parité de N	Réponse impulsionnelle symétrique	Réponse impulsionnelle anti-symétrique
N impair (N = 2n+1)	$h(n)+2.\sum_{k=1}^{n} h(n-k).\cos(2\pi kf)$	$2.\sum_{k=1}^{n} h(n-k).\sin(2\pi kf)$
N pair (N = 2n)	$2.\sum_{k=1}^{n} h(n-k).\cos(2\pi(k-1/2)f)$	$2.\sum_{k=1}^{n} h(n-k).\sin(2\pi(k-1/2)f)$

Tableau II-2-Expressions de D(f) et de W(f) du problème équivalent pour les quatre cas.

Symétrie → / ↓ Parité de N	Réponse impulsionnelle symétrique	Réponse impulsionnelle anti-symétrique
N impair N = 2n+1	$\hat{D}(f)=D(f)$ $\hat{W}(f)=W(f)$ $\begin{cases} h(i)=d_{n-i}/2 \\ \quad i=0,\ldots,n-1 \\ h(n)=d(0) \end{cases}$	$\hat{D}(f)=D(f)/\sin(2\pi f)$ $\hat{W}(f)=W(f).\sin(2\pi f)$ $\begin{cases} h(i)=d_{n-i}/2 \\ \quad i=0,\ldots,n-1 \\ h(n)=0 \end{cases}$
N pair N = 2n	$\hat{D}(f)=D(f)/\cos(\pi f)$ $\hat{W}(f)=W(f).\cos(\pi f)$ $\begin{cases} h(i)=d_{n-i}/2 \\ \quad i=0,\ldots,n-1 \end{cases}$	$\hat{D}(f)=D(f)/\sin(\pi f)$ $\hat{W}(f)=W(f).\sin(\pi f)$ $\begin{cases} h(i)=d_{n-i}/2 \\ \quad i=0,\ldots,n-1 \end{cases}$

II-3-METHODE UTILISANT L'ALGORITHME DE REMEZ

L'algorithme de Remez [3,4,5,12-21] est une procédure itérative qui fournit, après un nombre d'itérations fini, les valeurs des coefficients h(i) optimisant la fonction erreur E(f) au sens du minimax (norme de Chebyshev).

La théorie d'approximation de Chebyshev [13,14,15] montre que le problème formulé dans (II-1) admet une solution unique. La condition nécessaire et suffisante qui caractérise cette solution est donnée par le théorème de l'alternance suivant.

II-3-1-Théorème de l'alternance [12]

Soit $\mathcal{F}$ une partie fermée de l'intervalle [0,1/2]. Pour que la fonction G(f) donnée par l'équation (II-4) soit la meilleure approximation de D(f) (au sens du minimax), il est nécessaire et suffisant que la fonction erreur E(f) définie par:

$$E(f) = W(f).(G(f)-D(f)) \qquad \text{(II-6)}$$

présente sur $\mathcal{F}$ au plus n+2 alternances (extréma) telles que:

$$E(f_k)=-E(f_{k-1})=\pm\mathcal{E} \ , \quad f_k \in \mathcal{F} \ , k=1,\ldots,n+1 \qquad \text{(II-7)}$$

où

$$\mathcal{E} = \max_{f\in\mathcal{F}} |E(f)| . \qquad \text{(II-8)}$$

Les fréquences qui correspondent à ces extréma sont appelés fréquences extrémales.

II-3-2-Procédure de calcul de l'algorithme de Remez

Cette procédure est décrite par l'organigramme de la figure II-1. L'algorithme commence par le choix d'un ensemble de n+2 fréquences arbitraires formant ainsi une référence de fréquences extrémales de départ et converge vers la solution optimale après un certain nombre de pas.

Au ième pas de l'algorithme [12], il y a n+2 fréquences $\{f_k, i=0,\ldots,n+1\}$ pour lesquelles l'erreur E(f) est forcée à avoir l'amplitude $\mathcal{E}$ avec des signes alternés. Cela requiert la résolution de n+2 équations du type:

$$W(f_k).(D(f_k)-H(f_k)) = -(-1)^k \mathcal{E} \ , \quad k=0,\ldots,n+1 \qquad \text{(II-9)}$$

et qui peuvent être résumées par le système (II-10).

Ce système étant non singulier [12], il admet une solution unique pour $\mathcal{E}$ et les variables d_i. Cependant, la résolution directe de ce système est trop lourde (surtout si n est grand).

$$\begin{bmatrix} 1 & \cos(2\pi f_0) & \dots & \cos(2\pi n f_0) & 1/W(f_0) \\ 1 & \cos(2\pi f_1) & \dots & \cos(2\pi n f_1) & (-1)/W(f_1) \\ \cdot & \cdot & & \cdot & \cdot \\ \cdot & \cdot & & \cdot & \cdot \\ \cdot & \cdot & & \cdot & \cdot \\ 1 & \cos(2\pi f_{n+1}) & \dots & \cos(2\pi n f_{n+1}) & (-1)^{n+1}/W(f_{n+1}) \end{bmatrix} \begin{bmatrix} d_0 \\ d_1 \\ \cdot \\ \cdot \\ d_n \\ \varepsilon \end{bmatrix}$$

$$= \begin{bmatrix} D(f_0) \\ D(f_1) \\ \cdot \\ \cdot \\ \cdot \\ D(f_{n+1}) \end{bmatrix} \qquad \text{(II-10)}$$

Il est , par contre, très intéressant de calculer tout d'abord la quantité ε d'une manière analytique par la relation suivante:

$$\varepsilon = \frac{a_0 D(f_0) + a_1 D(f_1) + \dots a_{n+1} D(f_{n+1})}{a_0/W(f_0) - a_1/W(f_1) + \dots + (-1)^{n+1} a_{n+1}/W(f_{n+1})} \qquad \text{(II-11)}$$

où

$$a_k = \prod_{\substack{i=0 \\ i \neq k}}^{n+1} \frac{1}{(x_i - x_k)} \qquad \text{(II-12)}$$

avec

$$x_k = \cos(2\pi f_k) \qquad \text{(II-13)}$$

A l'étape suivante, La formule d'interpolation de Lagrange est utilisée pour calculer la réponse fréquentielle du filtre. En effet, la fonction H(f) est interpolée sur les n+1 points $f_0,\dots,f_n$ aux valeurs:

$$c_k = D(f_k) - (-1)^k \frac{\varepsilon}{W(f_k)} \qquad k=0,\dots,n \qquad \text{(II-14)}$$

on a donc

$$H(f) = \frac{\sum_{k=0}^{n} [\alpha_k/(x-x_k)]c_k}{\sum_{k=0}^{n} [\alpha_k/(x-x_k)]} \qquad (II-15)$$

où

$$\alpha_{k_0} = \prod_{\substack{i=0 \\ i\neq k}}^{n} \frac{1}{(x_i-x_k)} \qquad (II-16)$$

La fonction erreur E(f) est alors évaluée par la relation (II-6) pour tout f de l'intervalle $\mathcal{F}$. Cependant, il n'est nécessaire d'évaluer E(f) que sur un nombre fini de points; une grille de 16×n points régulièrement espacés s'avère suffisamment dense [11].

La dernière étape consiste à vérifier si $|E(f)| \leq \delta$ pour tout $f \in \mathcal{F}$. Si tel est le cas, l'optimum est atteint et G(f) est la meilleure approximation de D(f) sur $\mathcal{F}$.

Si, par contre, on a $|E(f)| > \delta$ pour quelques valeurs de $f \in \mathcal{F}$, on choisit un autre ensemble de n+2 fréquences constituant ainsi une nouvelle référence de fréquences extrémales. Pour assurer la convergence, on choisit ces nouvelles fréquences de telle manière que δ décroit à l'itération suivante. Si les nouveaux points sont choisis comme étant les fréquences qui correspondent aux extréma de E(f), δ se trouve alors forcé de décroître [12] et converge vers une valeur minimale qui correspond à la solution optimale du problème.

II-4-METHODE UTILISANT L'ALGORITHME DU SIMPLEXE

II-4-1-Description générale

L'algorithme du simplexe fut découvert par G.B. Dantzig en 1947. Il représente certainement une étape décisive dans l'histoire de l'optimisation combinatoire.

L'algorithme du simplexe [7,22-25] est une procédure d'optimisation d'une certaine quantité z, appelée *fonction coût* (ou *fonction économique*), combinaison linéaire d'une ou de plusieurs variables x_i, i=1 jusqu'à s, soumises à des contraintes se présentant comme des équations ou inéquations linéaires de ces mêmes variables.

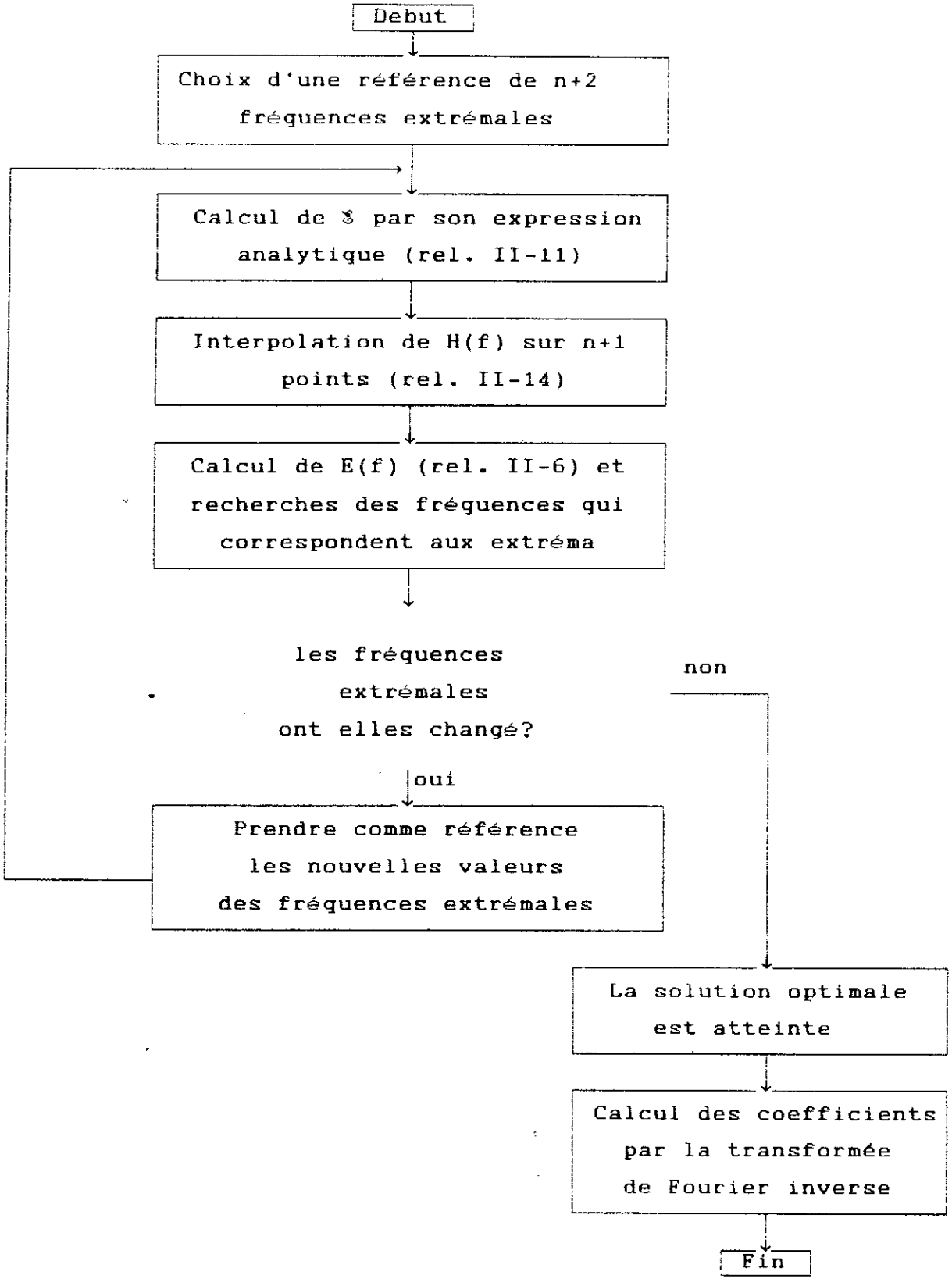


Fig.II-1- Algorithme de Remez.

L'algorithme du simplexe est une méthode de résolution d'un problème de programmation linéaire dont la formulation générale est la suivante:

$$\begin{cases} \underset{(x_i)}{\text{Min}}\ z=c^t x, \quad x=(x_1,x_2,\ldots,x_s)^t \\ A.x \le b \\ x_i \ge 0 \qquad i=1,\ldots,s \end{cases} \tag{II-17}$$

où, c et b sont des vecteurs colonnes, A est une matrice rxs, (r et s sont, respectivement, le nombre de contraintes sur les variables x_i et le nombre de ces variables).

Le problème ainsi défini est résumé dans un tableau appelé *tableau simplexe*. L'algorithme du simplexe opère sur ce tableau des transformations successives pour aboutir, à la fin, à la solution optimale si celle-ci existe.

Un développement détaillé sur l'algorithme du simplexe est présenté dans l'annexe I.

II-4-2-Formulation du problème d'optimisation minimax des filtres RIF à phase linéaire sous forme d'un programme linéaire

Nous allons montrer qu'il est possible de mettre le problème de minimax de Chebychev formulé dans (II-1) sous forme d'un programme linéaire [1,7] qui peut être résolu par l'algorithme du simplexe.

Soit $\mathcal{E}$ la valeur maximale de la fonction erreur E(f) à optimiser, telle que:

$$\|E(f)\| \le \mathcal{E} \qquad \forall\ f \in \mathcal{F} \tag{II-18}$$

Remplaçons G(f) par son expression générale (II-4) dans l'expression de l'erreur donnée par (II-6). Il vient:

$$E(f) = W(f).\left(\sum_{i=0}^{n} d_i.\cos(2\pi i f) - D(f)\right) \tag{II-19}$$

Introduisons maintenant cette expression dans la relation (II-18), nous pouvons écrire alors:

$$\begin{cases} \sum_{i=0}^{n} d_i.\cos(2\pi i f) - \mathcal{E}/W(f) \le D(f) \\ -\sum_{i=0}^{n} d_i.\cos(2\pi i f) - \mathcal{E}/W(f) \le -D(f) \end{cases} \tag{II-20}$$

La condition de positivité, nécessaire pour l'application de l'algorithme du simplexe (cf.Annexe I), nous conduit à faire le changement de variables suivant:

$$x_i = d_i + a \qquad i=0,1,\ldots,n \tag{II-21}$$

où a est une constante arbitraire à choisir suffisamment grande pour s'assurer que toutes les variables x_i sont positives. Sachant que pour les filtres non amplificateurs on a toujours [11]:

$$|h_i| \leq 1, \qquad i=0,1,\ldots,n \tag{II-22}$$

et en tenant compte des relations du tableau II-2 entre les variables h(i) et d_i, nous pouvons affirmer qu'une valeur de 10 pour la constante a s'avère satisfaisante. La relation (II-21) est utilisée pour retrouver, après optimisation, les coefficients d_i à partir des coefficients x_i.

Compte tenu de ce changement, nous pouvons écrire:

$$\begin{cases} \sum_{i=0}^{n} x_i .\cos(2\pi if) - \delta/W(f) \leq D(f) + a\sum_{i=0}^{n}\cos(2\pi if) & \text{(II-23-a)} \\ -\sum_{i=0}^{n} x_i .\cos(2\pi if) - \delta/W(f) \leq -D(f) - a\sum_{i=0}^{n}\cos(2\pi if) & \text{(II-23-b)} \\ x_i \geq 0, \quad i=0,\ldots,n \end{cases}$$

Ces deux inégalités doivent être vérifiées pour tout $f\in\mathcal{F}$. On procède alors à l'échantillonnage du domaine fréquentiel aux points $\{f_k,\ k=0,1,\ldots,Ne\}$. Le pas d'échantillonnage doit être suffisamment petit pour que l'analyse numérique de ce problème soit efficace. L'expérience montre que le nombre d'échantillons Ne doit être choisi proportionnel à n, soit

$$Ne = Lgridxn \tag{II-24}$$

Le facteur Lgrid est pris, en général, égal à 16 [11].

Il est à noter que δ est traité ici comme une variable et la fonction coût à optimiser est donnée par:

$$z=\delta \tag{II-25}$$

Ce problème peut être formulé sous forme d'un problème de programmation linéaire. Posons:

$$c^t=(0,0,\ldots,0,1) \tag{II-26}$$

$$x^t=(x_0,x_1,\ldots,x_n,\delta) \tag{II-27}$$

et soient A la matrice définie par les éléments suivants:

$$A = \begin{bmatrix} 1 & 2\cos(2\pi f_0) & \dots & 2\cos(2\pi(n-1)f_0) & 2\cos(2\pi n f_0) & -1/W(f_0) \\ 1 & 2\cos(2\pi f_1) & \dots & 2\cos(2\pi(n-1)f_1) & 2\cos(2\pi n f_1) & -1/W(f_1) \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ 1 & 2\cos(2\pi f_{Ne}) & \dots & 2\cos(2\pi(n-1)f_{Ne}) & 2\cos(2\pi n f_{Ne}) & -1/W(f_{Ne}) \\ -1 & -2\cos(2\pi f_0) & \dots & -2\cos(2\pi(n-1)f_0) & -2\cos(2\pi n f_0) & -1/W(f_0) \\ -1 & -2\cos(2\pi f_1) & \dots & -2\cos(2\pi(n-1)f_1) & -2\cos(2\pi n f_1) & -1/W(f_1) \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ -1 & -2\cos(2\pi f_{Ne}) & \dots & -2\cos(2\pi(n-1)f_{Ne}) & -2\cos(2\pi n f_{Ne}) & -1/W(f_{Ne}) \end{bmatrix}$$

et b le vecteur colonne défini par les éléments:

$$b = \begin{bmatrix} D(f_0) + a\sum_{i=0}^{n}\cos(2\pi i f_0) \\ D(f_1) + a\sum_{i=0}^{n}\cos(2\pi i f_1) \\ \vdots \\ D(f_{Ne}) + a\sum_{i=0}^{n}\cos(2\pi i f_{Ne}) \\ -D(f_0) - a\sum_{i=0}^{n}\cos(2\pi i f_0) \\ -D(f_1) - a\sum_{i=0}^{n}\cos(2\pi i f_1) \\ \vdots \\ -D(f_{Ne}) - a\sum_{i=0}^{n}\cos(2\pi i f_{Ne}) \end{bmatrix}$$

Le problème d'optimisation formulé sous les contraintes (II-23-a) et (II-23-b) peut être alors réécrit sous la forme condensée suivante:

$$\begin{cases} \min z = c^t x \\ A.x \le b \\ x \ge 0 \end{cases} \qquad \text{(II-28)}$$

qui est bien conforme au problème de programmation linéaire formulée dans (II-17).

Nous pouvons remarquer qu'il est possible d'introduire d'autres contraintes dans la matrice A. Nous verrons plus loin (Chap IV) l'importance capitale de cette option propre à cette méthode dans l'application de calcul des filtres numériques avec contraintes supplémentaires.

II-5-COMPARAISON ENTRE LA METHODE DE REMEZ ET LA METHODE DU SIMPLEXE

Nous présentons ici les résultats comparatifs obtenus en appliquant les deux méthodes sur les deux exemples définis dans le chapitre I et qui sont donnés par les gabarits des figures I-1 et I-2 (cf. § I-2-2).

Pour la méthode de Remez, nous avons utilisé le programme élaboré par JAMES H. Mc CLELLAN et THOMAS W. PARKS (1973) [5]. Tandis que pour la méthode du simplexe, nous avons développé un programme en Fortran 77 sur la base des données du problème (voir détails au chapitre III). La machine utilisée est un VAX/VMS 785 de Digital Equipment Corporation.

Les résulats obtenus qui sont présentés dans les figures II-2 et II-3 et les tableaux II-3 et II-4 sont identiques. Cependant, ces deux procédures diffèrent par leur temps d'exécution.

La courbe de la figure II-4 représente le temps d'exécution des deux programmes en fonction de N (ordre du filtre). Nous pouvons constater que la deuxième méthode est beaucoup plus lente par rapport à la première.

Cela s'explique, théoriquement, par le fait que le temps d'exécution de l'algorithme de Remez varie d'une façon quadratique en fonction de n (c-à-d $O(n^2)$) tandis que celui de l'algorithme du simplexe varie d'une façon exponentielle en fonction de r (c-à-d $O(\alpha^r)$) où α est une constante et r est le rang de la matrice des contraintes A et qui est égal à 2×Ne soit 2×16×n.

Dans la figure II-4 les estimations confirment ces prévisions théoriques (les courbes d'interpolation sont représentées en trait continu). Pour le cas du filtre numérique passe-bas défini par le gabarit de la figure I-1, les temps d'exécution en fonction de N estimés pour la méthode de Remez et la méthode du simplexe sont respectivement:

$$t_1(N) \simeq 0,2N^{1,8}$$

et

$$t_2(N) \simeq 100\exp(0,213N)$$

II-6-OPTIMISATION AVEC CONTRAINTES SUPPLEMENTAIRES

Les réalisations pratiques sont souvent soumises à des contraintes supplémentaires dont il faut tenir compte lors du calcul. La méthode du simplexe peut servir de base pour résoudre ce genre de problème. On est appelé ainsi, soit à introduire ces mêmes contraintes mises sous forme d'équations ou d'inéquations dans le système linéaire initial, soit à résoudre ce dernier et manipuler les coefficients obtenus de telle manière à ce que les contraintes supplémentaires soient respectées ainsi que les contraintes principales. Cette dernière procédure est plus rapide que la première méthode, mais, on assiste dans ce cas à une dégradation plus ou moins importante de l'option à optimiser qui est la norme de l'erreur maximale.

Le choix de l'une ou l'autre des deux méthodes est donc lié aux deux paramètres - souvent contradictoires - qui sont le temps d'exécution et l'erreur maximale.

Nous développerons plus loin (chap.IV) un exemple de contraintes sur la réponse indicielle souvent exigées dans la réalisation des filtres numériques.

II-7-CONCLUSION

La conclusion principale que nous pouvons tirer de cette étude est que la méthode de Remez est beaucoup plus rapide que la méthode du simplexe. Cependant, cette dernière offre une propriété très intéressante qui est la possibilité de pouvoir introduire des contraintes supplémentaires dans le problème d'optimisation. Cet avantage nous a poussé à améliorer cette méthode en vu de réduire son temps d'exécution (voir chapitre suivant).

Tableau II-3-Coefficients du filtre numérique passe-bas défini par le gabarit de la figure I-1 pour N=33.

	Méthode de Remez	Métohde du Simplexe
$h_0 = h_{32} =$	-0.0058	-0.0063
$h_1 = h_{31} =$	0.0125	0.0127
$h_2 = h_{30} =$	0.0111	0.0117
$h_3 = h_{29} =$	-0.0049	-0.0045
$h_4 = h_{28} =$	-0.0146	-0.0148
$h_5 = h_{27} =$	0.0029	0.0053
$h_6 = h_{26} =$	0.0223	0.0225
$h_7 = h_{25} =$	0.0042	0.0048
$h_8 = h_{24} =$	-0.0305	-0.0301
$h_9 = h_{23} =$	-0.0175	-0.0181
$h_{10} = h_{22} =$	0.0378	0.0375
$h_{11} = h_{21} =$	0.0414	0.0419
$h_{12} = h_{20} =$	-0.0442	-0.0436
$h_{13} = h_{19} =$	-0.0917	-0.0921
$h_{14} = h_{18} =$	0.0484	0.0477
$h_{15} = h_{17}$	0.3133	0.3134
$h_{16} =$	0.4501	0.4508
Erreur	0.0206	0.0213
Temps	0.5 s	605 s

Tableau II-4-Coefficients du filtre numérique passe-bande défini par le gabarit de la figure I-2 pour N=33.

	Méthode de Remez	Méthode du Simplexe
$h_0 = h_{32} =$	0.0044	0.0030
$h_1 = h_{31} =$	-0.0025	-0.0037
$h_2 = h_{30} =$	0.0066	0.0075
$h_3 = h_{29} =$	-0.0274	-0.0269
$h_4 = h_{28} =$	-0.0114	-0.0113
$h_5 = h_{27} =$	0.0214	0.0212
$h_6 = h_{26} =$	-0.0007	-0.0009
$h_7 = h_{25} =$	0.0294	0.0300
$h_8 = h_{24} =$	-0.0182	-0.0178
$h_9 = h_{23} =$	-0.0648	-0.0645
$h_{10} = h_{22} =$	0.0321	0.0312
$h_{11} = h_{21} =$	-0.0003	-0.0002
$h_{12} = h_{20} =$	0.0071	0.0071
$h_{13} = h_{19} =$	0.0889	0.0892
$h_{14} = h_{18} =$	-0.2839	-0.2836
$h_{15} = h_{17}$	-0.0586	-0.0586
$h_{16} =$	0.4005	0.4000
Erreur	0.0265	0.0269
Temps	0.6 s	721 s

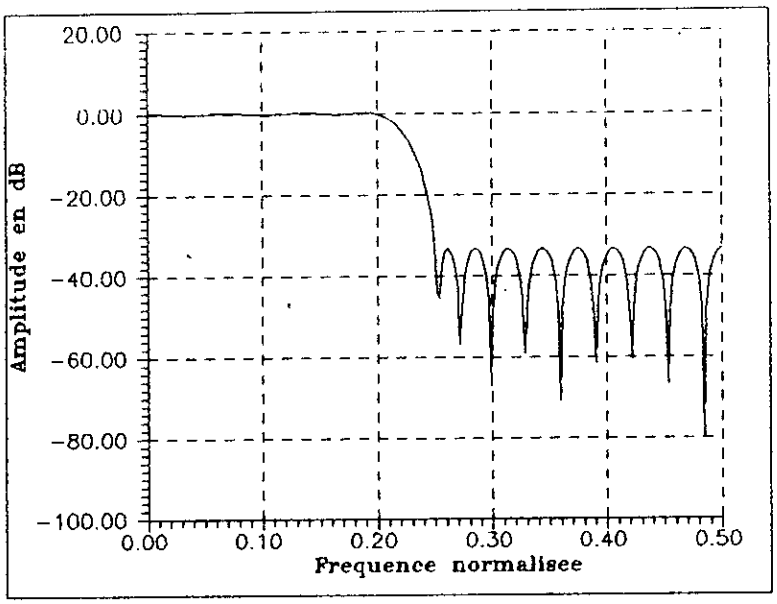


Fig.II-2-Spectre d'amplitude d'un filtre numerique passe-bas d'ordre 33 defini par le gabarit de la figure I-1.

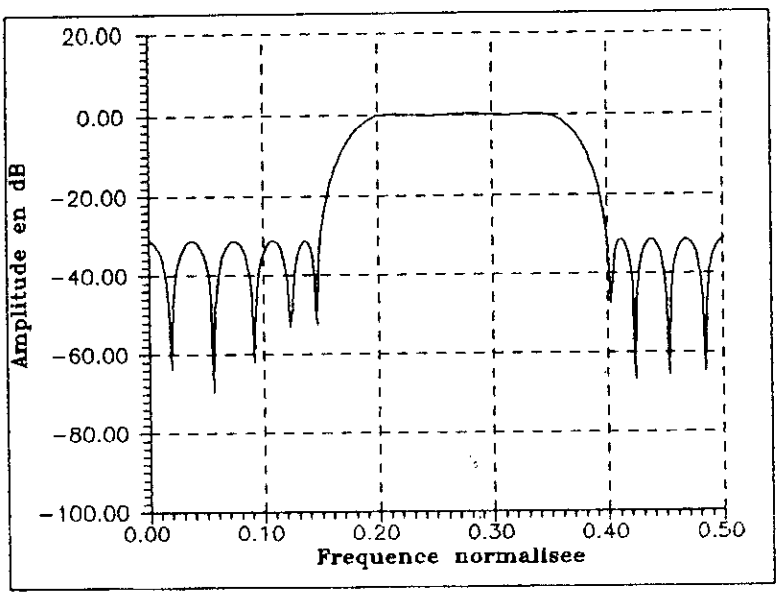


Fig.II-3-Spectre d'amplitude d'un filtre numerique pass-band d'ordre 33 defini par le gabarit de la figure I-2.

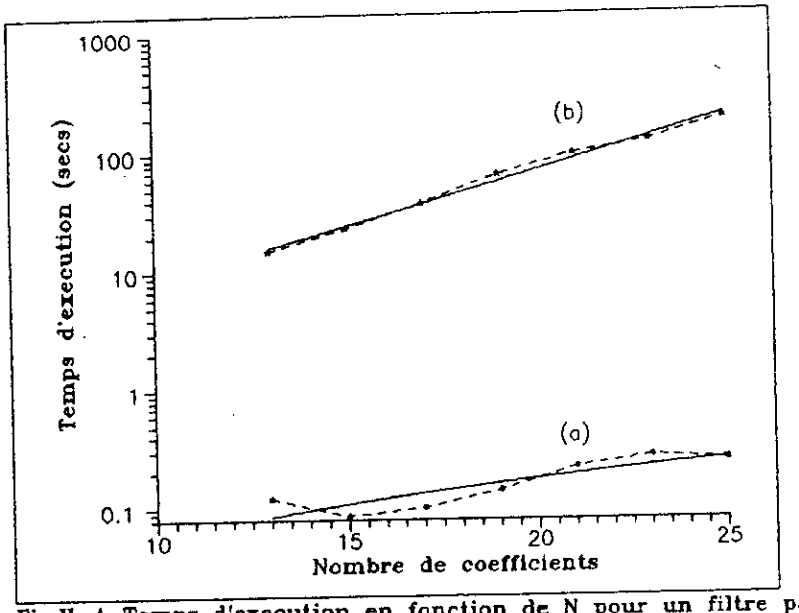


Fig.II-4-Temps d'execution en fonction de N pour un filtre passe-bas (les courbes d'interpolation sont representees en trait plein):
(a)-Methode de Remez
(b)-Methode du simplexe

CHAPITRE III

UNE METHODE AMELIOREE POUR L'OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE UTILISANT L'ALGORITHME DU SIMPLEXE

III-1-INTRODUCTION

L'analyse comparative entre les deux méthodes d'optimisation des filtres numériques au sens de Chebyshev (Chap.II) à savoir la méthode de Remez et la méthode du simplexe nous a conduit à la conclusion suivante : Bien que la première méthode soit beaucoup plus rapide que la deuxième, cette dernière conserve toujours son avantage de permettre l'introduction de contraintes supplémentaires dans le problème initial. Or, cette possibilité est d'une importance capitale dans le sens où il est nécessaire d'envisager des solutions spécifiques à des problèmes particuliers.

Notre travail a pour objectif d'améliorer le temps de convergence de la méthode du simplexe tout en maintenant son avantage principal cité plus haut. Nous allons présenter, dans ce chapitre, une méthode améliorée basée sur l'algorithme du simplexe. Les résultats seront comparés aux résultats obtenus par l'ancienne méthode du simplexe. L'optimisation avec adjonction de contraintes supplémentaires aux contraintes fréquentielles sera présentée dans le prochain chapitre (Chap. IV).

III-2-PRESENTATION DE LA METHODE DU SIMPLEXE AMELIOREE

Nous avons vu (§ II-4) que l'utilisation de la méthode du simplexe nécessite la discrétisation du domaine fréquentiel en 16×n points. L'optimisation se fait alors, en une seule itération, en prenant tous les échantillons de la réponse désirée D(f) et de la fonction de pondération W(f). Le temps d'exécution de la procédure simplexe est alors extrêmement grand. En modifiant cette méthode nous pouvons obtenir une nouvelle procédure beaucoup plus rapide. Cette nouvelle méthode du simplexe s'inspire du théorème de l'alternance (§ II-3-1). Elle repose sur le principe suivant: La fonction erreur E(f) présente à l'optimum un nombre fini

d'extréma pour un nombre de coefficients donné (n+2 extréma pour 2n+1 coefficients) et les fréquences extrémales correspondant à ces extréma prennent des positions fixes sur l'axe fréquentiel. L'idée est la suivante:

1^{ere} étape: nous choisissons n+2 fréquences uniformément réparties sur $\mathcal{F}$ destinées à être les fréquences extrémales de départ.

2^{eme} étape: nous résolvons le problème par l'algorithme du simplexe en prenant comme échantillons les n+2 fréquences extrémales de départ (au lieu de 16×n pour l'ancienne méthode du simplexe). Les coefficients obtenus sont utilisés pour évaluer la fonction erreur E(f). Nous déterminons ensuite les nouvelles fréquences extrémales qui correspondent aux extréma de E(f).

3^{eme} étape: Nous comparons les nouvelles fréquences extrémales avec les fréquences de départ en nombre et en positon. Si le nombre est le même et si l'écart est suffisamment faible, nous pouvons dire que l'optimum est atteint, sinon, les nouvelles fréquences sont prises comme fréquences extrémales de départ et nous revenons à la 2^{eme} étape.

A la différence de l'ancienne méthode du simplexe, la méthode du simplexe améliorée est exécutée en plusieurs itérations.

Cette nouvelle méthode peut être représentée par l'organigramme de la figure III-1.

III-3-ELABORATION D'UN PROGRAMME ET RESULTATS

III-3-1-Description du programme

Nous avons élaboré un programme (FASTSIMPLEX) en Fortran 77 traduisant l'organigramme de la figure III-1. La formulation du problème d'optimisation étant la même que pour l'ancienne méthode du simplexe, il convient alors de se referer au paragraphe II-4 car nous avons gardé les mêmes notations. Le programme que nous avons écrit contient les étapes suivantes:

1-Lecture des données

Les données principales sont le gabarit et l'ordre du filtre. Le gabarit est lu directement à partir d'un fichier qui doit contenir le nombre de bandes passantes et coupées, NBAND, et pour chaque bande, les bornes de celle-ci, la valeur désirée D et la

pondération W (qui sont constantes par bande). L'ordre du filtre NFILT est ensuite demandé de l'utilisateur.

2-Discrétisation du problème

A partir des valeurs prises par D et W dans chaque bande, nous définissons des fonctions échantillonnées DN et WN qui prennent les valeurs de D et W prélevées à des valeurs discrètes de la fréquence et qui sont au nombre de 16×n (n=(NFILT-1)/2 dans le cas où NFILT est impair et n=NFILT/2 si NFILT est pair) réparties uniformément sur l'axe fréquentiel. Ces fréquences discrètes sont désormais désignées par des nombres entiers compris entre 0 et 16xn. La densité Lgrid (cf. § II-4-2) fixée à 16 dans le programme peut être modifiée ou introduite comme un paramètre à fixer au départ.

3-Formulation d'un problème équivalent au problème initial

Nous avons montré (cf.§ II-2) qu'il est possible d'envisager quatre types de filtres RIF à phase linéaire et qu'il est possible de traiter un problème unique à condition de respecter les règles de changement du tableau II-2.

Pour le cas de notre programme, nous avons retenu deux cas: le cas où NFILT est impair et le cas où il est pair, la symétrie étant toujours prise positive.

4-Initialisation des fréquences extrémales

Nous prélevons n+2 fréquences extrémales uniformément réparties sur l'intervalle $\mathcal{F}$ (ce qui revient à prendre n+2 valeurs entières uniformément réparties sur l'intervalle [0,16×n]). Par mesure de sécurité, parmi les fréquences extrémales de départ nous prenons aussi les fréquences qui correspondent aux frontières des bandes passantes et coupées car, c'est en ces points que l'erreur est soupçonnée être la plus importante. Le nombre des fréquences extrémales (NEXT) de départ est, par conséquent, légèrement supérieur à n+2.

5-construction du premier tableau Simplexe (Voir Annexe I)

Soit {NFEXT(I), I=1,...,NEXT} l'ensemble des fréquences extrémales de départ. L'application du simplexe nécessite la mise

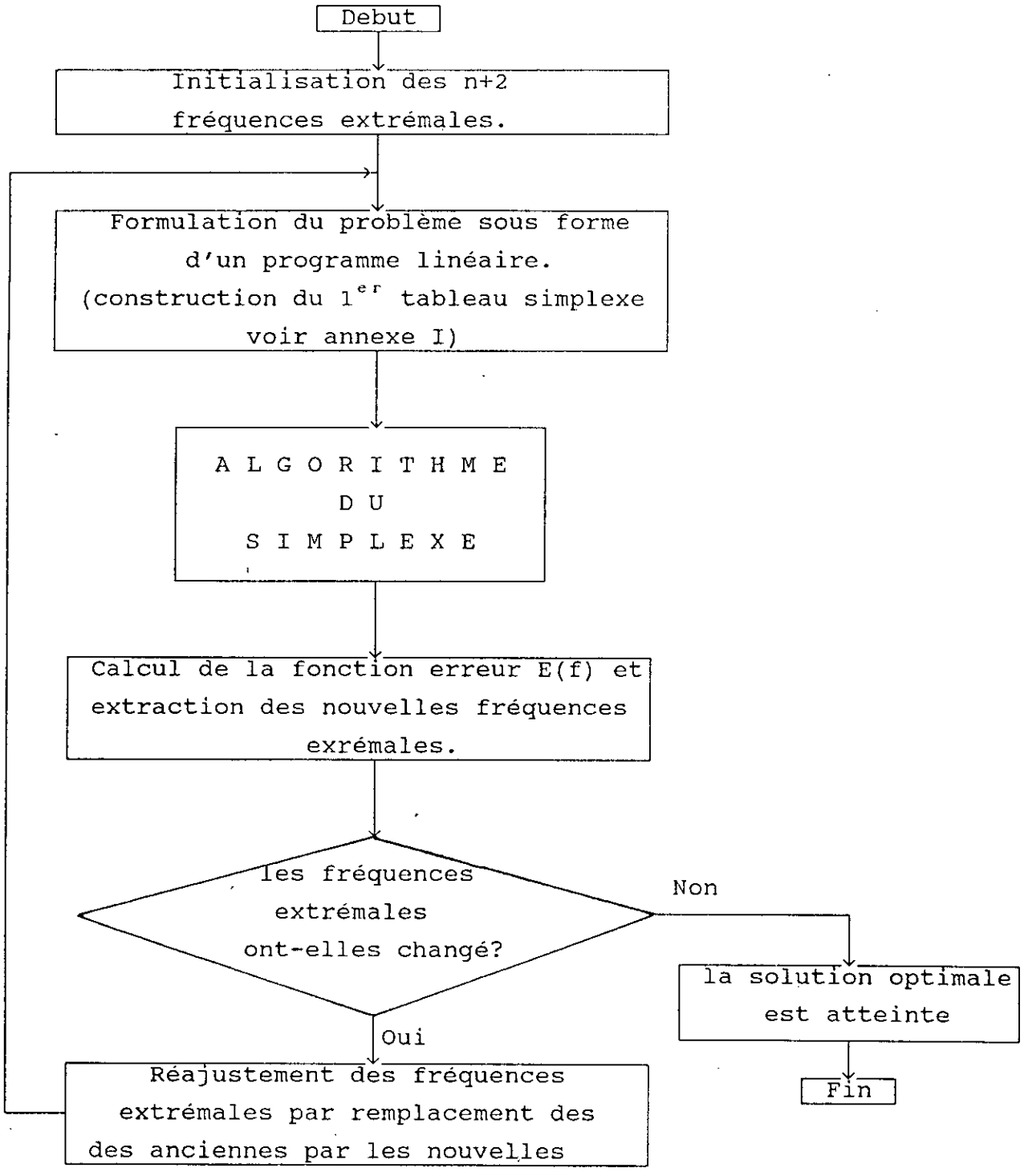


Fig.III-1-Méthode du simplexe améliorée.

des contraintes sous forme d'égalités en introduisant dans le problème linéaire des variables d'écart (cf. Annexe I). Dans ce type de problème il faut ajouter autant de variables d'écart qu'il y a de contraintes. La matrice A se trouve donc augmentée de r colonnes, où r est le nombre de contraintes. Nous pouvons donc formuler le problème comme suit:

$$\left[\begin{array}{c:c} & \begin{matrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \dots & \cdot \\ 0 & 0 & 0 & \dots & 1 \end{matrix} \end{array}\right] \quad \text{A} \quad .x = b \qquad \text{(III-1)}$$

où x est l'ensemble des variables structurelles et des variables d'écart.

Les variables structurelles et les variables d'écart sont traitées de la même manière et chacune d'elle est affectée d'un indice (de 1 à r+s). La répartition des indices est la suivante:
-Ensemble des variables structurelles: $\{x_j,\ j=1,2,\dots,s\}$.
-Ensemble des variables d'écart: $\{x_j,\ j=s+1,s+2,\dots,s+r\}$.

La matrice A, qui renferme les coefficients des variables structurelles dans les contraintes, est définie par les éléments suivants (cf.o II-4-2):

$$\begin{cases} \alpha_{ij} = \cos[2\pi.NFEXT(i).(j-1)] & \text{pour } i=1,2,\dots,NEXT \text{ et } j=1,2,\dots,s \\ \alpha_{i(n+1)} = -1/W(NFEXT(i)) & \text{pour } i=1,2,\dots,NEXT \end{cases} \qquad \text{(III-2-a)}$$

$$\begin{cases} \alpha_{ij} = -a_{(i-NEXT),j} & \text{pour } i=NEXT+1,\dots,2NEXT \text{ et } j=1,2,\dots,s \\ \alpha_{i(n+1)} = a_{(i-NEXT),(n+1)} & \text{pour } i=NEXT+1,\dots,2NEXT \end{cases} \qquad \text{(III-2-b)}$$

Le vecteur colonne b qui renferme les seconds membres des contraintes est défini par les éléments suivants:

$$\begin{cases} b_i = D[NFEXT(i)]+Q[NFEXT(i)] & \text{pour } i=1,2,\dots,NEXT \\ b_i = -b_{i-NEXT} & \text{pour } i=NEXT+1,\dots,2NEXT \end{cases} \qquad \text{(III-3)}$$

où

$$Q[NFEXT(i)] = a\sum_{j=i}^{s-1} \cos[2\pi.NFEXT(i).(j-1)] \qquad \text{(III-4)}$$

posons

$$r=2NEXT, \quad ITAB = r+1 \quad et \quad JTAB=r+s+1$$

Le premier tableau simplexe (ITABxJTAB) tel qu'il est défini dans l'annexe I, contient les éléments suivants:

-La première colonne, IB(i) (i=1,...,ITAB-1), contient les indices des variables de la solution de départ et la dernière colonne, $a_{i,JTAB}$(i=1,...,ITAB-1), contient les valeurs prises par ces variables (valeurs initiales du vecteur b).

-La matrice $(a_{i,j})$ (i=1,...,ITAB-1 et j=1,...,JTAB-1) contient les coefficients des variables (structurelles et d'écart) dans les contraintes et qui sont définis plus haut (relations (III-1) et (III-2)). On a:

$$a_{ij} = \begin{cases} \alpha_{ij} & \text{pour } i=1,...,ITAB-1 \text{ et } j=1,...,s \\ \delta_{i,(j-n)} & \text{ailleurs} \end{cases} \qquad (III-5)$$

où

$$\delta_{ij} = \begin{cases} 0 & si\ i \neq j \\ 1 & si\ i=j \end{cases} \qquad (III-6)$$

-La première ligne, C(j) (j=1,...,JTAB-1), contient les coefficients des variables dans la fonction économique et comme $z=\mathcal{E}$ nous avons alors:

$$C(j)=\begin{cases} 0 & \text{pour } j=1,...,JTAB \text{ et } j\neq s \\ 1 & \text{pour } j=s \end{cases} \qquad (III-7)$$

-La dernière ligne, $(x_{ITAB,j})$, j=1,...,JTAB-1, est initialisée aux valeurs suivantes:

$$a_{ITAB,j} = -C(j) \qquad j=1,...,JTAB-1 \qquad (III-8)$$

-Enfin, la quantité $x_{ITAB,JTAB}$ qui est la fonction économique $z=\mathcal{E}$ est initialisée à zéro.

Le chargement de la matrice $\{a_{ij}\}$ (pour i=1,...,ITAB-1 et j=1,...,JTAB-1) s'effectue à l'aide d'un sous-programme (SBR1) dont les paramètres d'entrée sont : le nombre des fréquences extrémales (NEXT), leurs valeurs (NFEXT(I), I=1,...,NEXT), la fonction objectif échantillonnée (DN) et la fonction de pondération échantillonnée (WN). A la sortie nous récupérons les éléments de la matrice $\{a_{ij}\}$.

6-Le noyau simplexe:

L'algorithme du simplexe direct nécessite l'obtention d'une solution réalisable de départ (cf.Annexe I). Pour éviter ce travail supplémentaire, nous avons utilisé l'algorithme dual-simplexe (Annexe I) qui ne nécessite que l'obtention d'une solution de base non réalisable pour laquelle la relation suivante est vérifiée:

$$a_{ITAB,j} \leq 0 \text{ pour } j = 1,\ldots,JTAB-1 \qquad \text{(III-9)}$$

Or, cette condition pour ce genre de problème est toujours satisfaite. En effet, il suffit de prendre comme solution de départ une base formée des seules variables d'écart. La solution de départ est donc:

$$\begin{cases} x_i = 0, \text{ pour } i = 1,\ldots,s \\ x_i = b_i, \text{ pour } i = s+1,\ldots,JTAB-1 \end{cases} \qquad \text{(III-10)}$$

Nous pouvons remarquer que la condition (III-9) est vérifiée dans le tableau simplexe initial (Eqt. (III-8)) et que les valeurs b_i ne sont pas toutes positives (dans le vecteur colonne b il y a autant de valeurs négatives que de valeurs positives), donc la condition de positivité (qui exige que toutes les variables soient positives) n'est pas respectée pour cette solution. Par conséquent, la solution envisagée n'est pas réalisable bien que la condition (III-9) soit vérifiée.

En toute rigueur, cette condition, vérifiée pour le problème principal et pour le problème soumis à des contraintes supplémentaires étudié au chapitre IV, est à réviser quand il s'agit de contraintes supplémentaires quelconques.

Le noyau simplexe est une traduction en Fortran 77 de l'algorithme dual-simplexe (Annexe I) mise sous forme d'un sous-programme (SBR2).

7-Evaluation et analyse de la fonction erreur

La fonction G(f) est évaluée par la relation (II-4). Les coefficients d_i dans l'expression (II-4) sont obtenus à partir des variables x_i, issus de l'étape précédente et solutions du problème d'optimisation, en utilisant la relation (II-21). La fonction erreur E(f), quant à elle, est évaluée, en tout point, par la

formule (II-6). L'analyse de cette expression nous permet de déterminer les extréma de celle-ci, et cela, est confié à un sous-programme (SBR3). On en déduit ainsi, les nouvelles fréquences extrémales en nombre (NEXT1) et en position (soient, NFEXT1(I), I=1,...,NEXT1).

8-Test de l'optimalité

Ce test consiste à comparer les nouvelles fréquences extrémales avec les anciennes. S'il n'y a pas de changement relativement à l'erreur tolérée, l'optimum global est atteint et G(f) est la meilleure approximation pour D(f), sinon, on revient à l'étape 5 (construction d'un nouveau tableau simplexe) pour une nouvelle séquence après avoir remplacé les anciennes fréquences extrémales par les nouvelles.

9-Section de sortie

Une fois la solution optimale est obtenue, le tableau II-2 est utilisé pour évaluer les coefficients du filtre numérique. Le programme donne aussi l'erreur optimale et le temps d'exécution.

Le programme FASTSIMPLEX est donné en annexe (ANNEXE II).

III-3-2-Résultats

Le test de cette méthode sur un nombre d'exemples très large a montré qu'elle converge toujours vers la solution optimale quel que soit le type du filtre. Ceci reste valable même en introduisant certaines contraintes supplémentaires comme nous pouvons le voir plus loin (Chap. IV). Nous présentons ici quelques exemples:

Exemple III-1: Filtre numérique passe-bas défini par le gabarit de la figure I-1 (cf. § I-2-2)). Il est testé pour plusieurs valeurs de l'ordre N.

Exemple III-2: Filtre numérique passe-bande défini par le gabarit de la figure I-2 (cf.§ I-2-2).

Exemple III-3: C'est un filtre passe-bande selectif défini par le gabarit suivant:

$$D(f)=\begin{cases} 0 & f \in [0.00,0.20] \\ 1 & f \in [0.21,0.26] \\ 0 & f \in [0.27,0.50] \end{cases} \quad \text{et} \quad W(f)=\begin{cases} 1 & f \in [0.00,0.20] \\ 1 & f \in [0.21,0.26] \\ 1 & f \in [0.27,0.50] \end{cases}$$

Exemple III-4: C'est un filtre numérique coupe-bande défini par le gabarit suivant :

$$D(f)=\begin{cases} 1 & f \in [0.00,0.15] \\ 0 & f \in [0.20,0.30] \\ 1 & f \in [0.35,0.50] \end{cases} \text{ et } W(f)=\begin{cases} 1 & f \in [0.00,0.15] \\ 1 & f \in [0.20,0.30] \\ 1 & f \in [0.35,0.50] \end{cases}$$

Exemple III-5: C'est un filtre numérique coupe-bande selectif défini par le gabarit suivant:

$$D(f)=\begin{cases} 1 & f \in [0.00,0.20] \\ 0 & f \in [0.21,0.26] \\ 1 & f \in [0.27,0.50] \end{cases} \text{ et } W(f)=\begin{cases} 1 & f \in [0.00,0.20] \\ 1 & f \in [0.21,0.26] \\ 1 & f \in [0.27,0.50] \end{cases}$$

Exemple III-6: C'est un filtre numérique passe-haut défini par le gabarit suivant:

$$D(f)=\begin{cases} 0 & f \in [0.00,0.20] \\ 1 & f \in [0.25,0.50] \end{cases} \text{ et } W(f)=\begin{cases} 1 & f \in [0.00,0.20] \\ 1 & f \in [0.25,0.50] \end{cases}$$

Exemple III-7: C'est un filtre numérique à 4 bandes défini par le gabarit suivant:

$$D(f)=\begin{cases} 0.5 & f \in [0.00,0.15] \\ 0.0 & f \in [0.17,0.27] \\ 1.0 & f \in [0.29,0.38] \\ 0.0 & f \in [0.40,0.50] \end{cases} \text{ et } W(f)=\begin{cases} 1 & f \in [0.00,0.15] \\ 1 & f \in [0.17,0.27] \\ 1 & f \in [0.29,0.38] \\ 1 & f \in [0.40,0.50] \end{cases}$$

Les spectres d'amplitudes des différents filtres définis ci-dessus et dont les coefficients sont optimisés par cette nouvelle méthode sont illustrées dans les figures III-2 à III-8. Le tableau III-1 donne, à titre de comparaison, les coefficients du filtre de l'exemple III-1 obtenus par l'ancienne méthode du simplexe et la méthode du simplexe améliorée.

III-3-3-Discussion du temps d'exécution

La courbe III-9, qui représente l'évolution du temps d'exécution en fonction du nombre de coefficients (pour un filtre numérique passe-bas), montre clairement que la méthode du simplexe améliorée, décrite plus haut, est beaucoup plus rapide que la méthode classique reposant sur l'algorithme du simplexe. La raison est la suivante: l'organe simplexe travaille ici sur une matrice

de rang 2×(n+2) au lieu de 2×16×n pour l'ancienne méthode. Le temps d'exécution d'une itération, qui varie d'une façon exponentielle en fonction du rang de la matrice des contraintes pour la procédure simplexe (cf. § II-5), se trouve donc énormément réduit.

En réalité, nous n'avons comparé, jusqu'ici, que le temps d'exécution d'une seule itération pour la nouvelle méthode avec le temps d'exécution de l'ancienne méthode (qui est exécutée en une seule itération). Si, I est le nombre d'itérations nécessaires pour atteindre la solution optimale, le temps d'exécution global de la nouvelle méthode est à peu près I fois le temps d'exécution d'une itération. Le nombre I, qui est imprévisible au départ, ne dépasse guère la valeur 10 pour les optimisations sans contraintes supplémentaires effectuées sur nos exemples (la tolérence sur les fréquences entières étant égale à l'unité). Ceci explique bien l'écart énorme observé dans la figure III-9. A titre d'exemple pour un filtre passe-bas d'ordre N=25 le rapport entre les temps d'exécution est de 20, une quantité vraiment appréciable.

Les estimations de l'évolution du temps d'exécution en fonction de N (ordre du filtre) pour l'ancienne méthode et la nouvelle méthode donnent respectivement (pour les courbes de la figure III-9):

$$t_1(N) \simeq 100 \exp(0{,}213N)$$

et

$$t_2(N) \simeq 239 \exp(0{,}059N)$$

ces résultats sont bien conformes aux prévisions théoriques.

Le test d'arrêt du programme est conditionné par l'invariance de l'ensemble des fréquences extrémales entières (appartenant au domaine [0,16×n]). Cette égalité stricte n'est pas aussi sévère que nous pouvons l'imaginer, car la comparaison des fréquences se fait sur les valeurs entières de celles-ci et l'erreur admise peut atteindre l'unité. En se ramenant dans l'échelle réelle des fréquences normalisées (c'est-à-dire l'intervalle [0,0.5]), l'erreur maximum admise est égale, dans ce cas, à 0.5/(16xn) dépendant de ce fait de l'ordre du filtre et de la densité de grille (optimisée à 16 pour un meilleur compromis temps d'exécution-précision). En diminuant cette dernière, il est possible de réduire le nombre d'itérations I et par conséquent, de réduire le temps d'exécution de la nouvelle méthode mais ceci

entraînerait une dégradation de la précision sur les résultats obtenus. Cela ne serait valable que pour un test grossier.

III-4-CONCLUSION

Nous avons présenté ici une méthode rapide pour l'optimisation des filtres RIF à phase linéaire qui repose sur l'algorithme du simplexe. Cette nouvelle méthode donne les mêmes résultats que ceux obtenus par l'ancienne méthode du simplexe avec un temps d'exécution nettemment amélioré.

Il va sans dire que le temps d'exécution de la méthode du simplexe améliorée reste toujours supérieur au temps d'exécution de la méthode de Remez, toutefois la procédure simplexe a l'avantage d'admettre d'autres contraintes dans le problème d'optimisation principal. Cette option sera étudiée au chapitre suivant.

Tableau III-1. Coefficients du filtre numérique RIF du type passe-bas défini par l'exemple III-1

	Ancienne méthode du simplexe	Nouvelle méthode du simplexe
$h_0 = h_{32} =$	-0.0063	-0.0065
$h_1 = h_{31} =$	0.0127	0.0117
$h_2 = h_{30} =$	0.0117	0.0118
$h_3 = h_{29} =$	-0.0045	-0.0045
$h_4 = h_{28} =$	-0.0148	-0.0147
$h_5 = h_{27} =$	0.0053	0.0056
$h_6 = h_{26} =$	0.0225	0.0227
$h_7 = h_{25} =$	0.0048	0.0050
$h_8 = h_{24} =$	-0.0301	-0.0302
$h_9 = h_{23} =$	-0.0181	-0.0182
$h_{10} = h_{22} =$	0.0375	0.0379
$h_{11} = h_{21} =$	0.0419	0.0418
$h_{12} = h_{20} =$	-0.0436	-0.0435
$h_{13} = h_{19} =$	-0.0921	-0.0925
$h_{14} = h_{18} =$	0.0477	0.0477
$h_{15} = h_{17}$	0.3134	0.3135
$h_{16} =$	0.4508	0.4510
Erreur	0.0211	0.0213
Temps	605 s	18 s

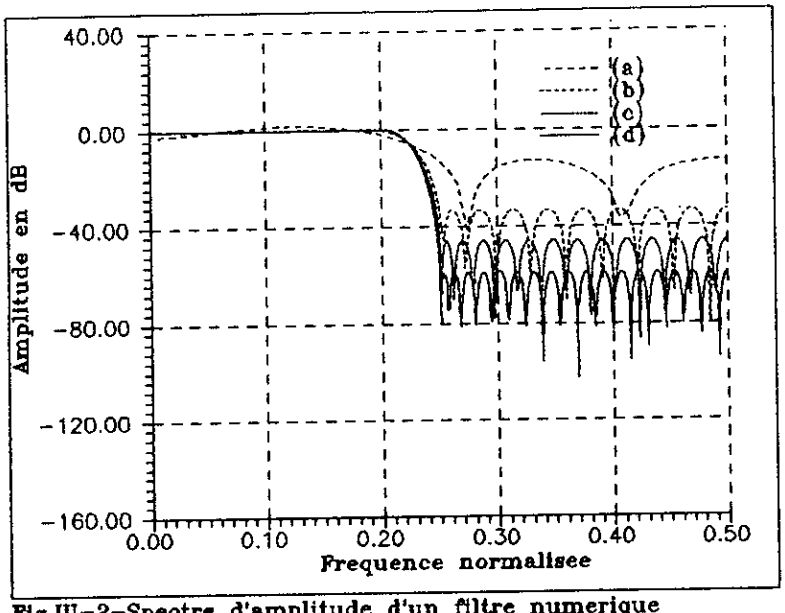


Fig.III-2-Spectre d'amplitude d'un filtre numerique passe-bas (exemple III-1) d'ordre: (a)-N=9, (b)-N=33, (c)-N=49, (d)-N=65

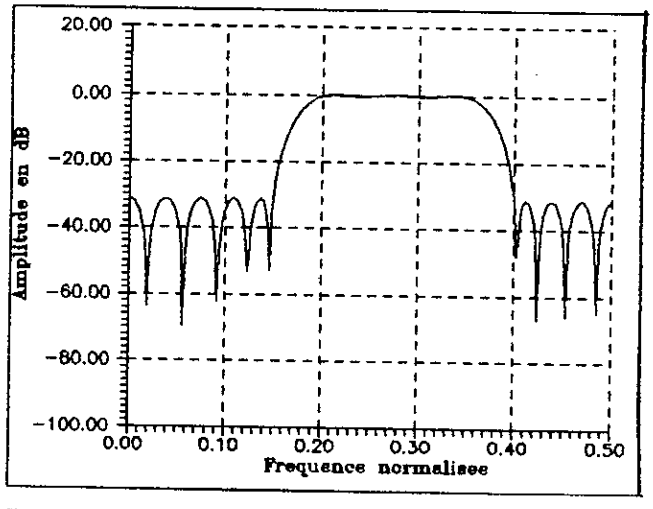


Fig.III-3-Spectre d'amplitude d'un filtre numerique passe-bande (exemple III-2) d'ordre 33.

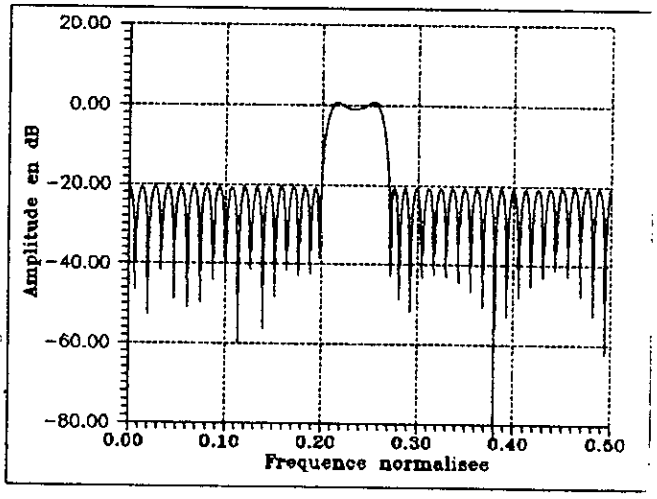


Fig.III-4-Spectre d'amplitude d'un filtre numerique passe-bande selectif (exemple III-3) d'ordre 81.

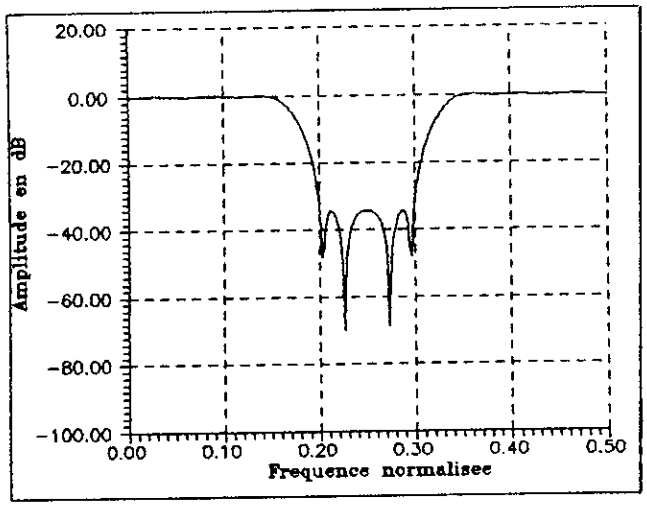


Fig.III-5-Spectre d'amplitude d'un filtre numerique coupe-bande (exemple III-4) d'ordre 33.

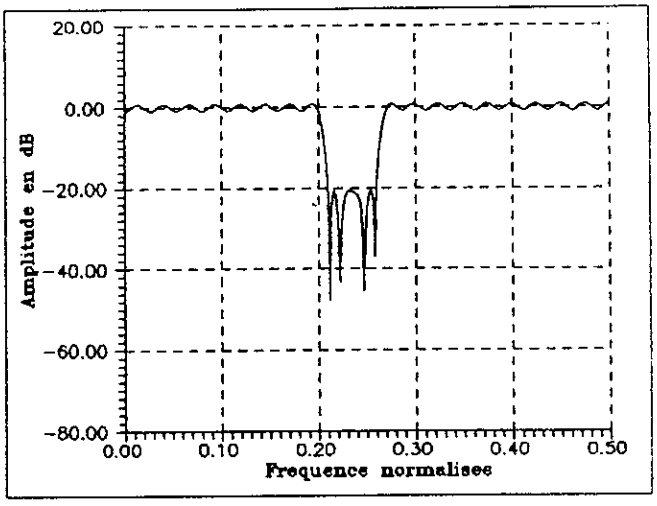


Fig.III-6-Spectre d'amplitude d'un filtre numerique coupe-bande selectif (exemple III-5) d'ordre 81.

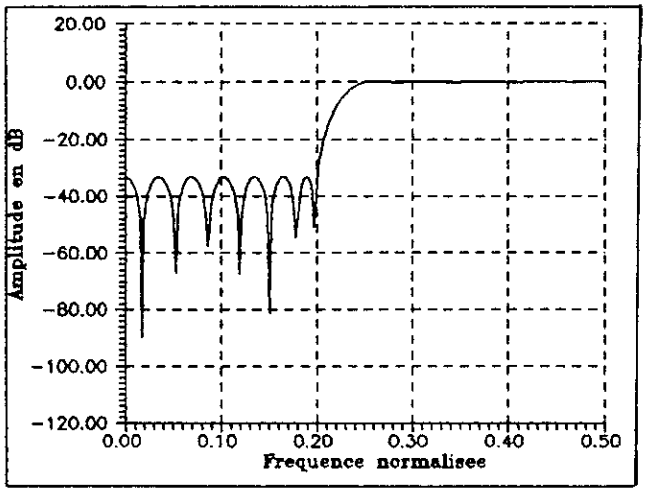


Fig.III-7-Spectre d'amplitude d'un filtre numerique passe-haut (exemple III-6) d'ordre 33.

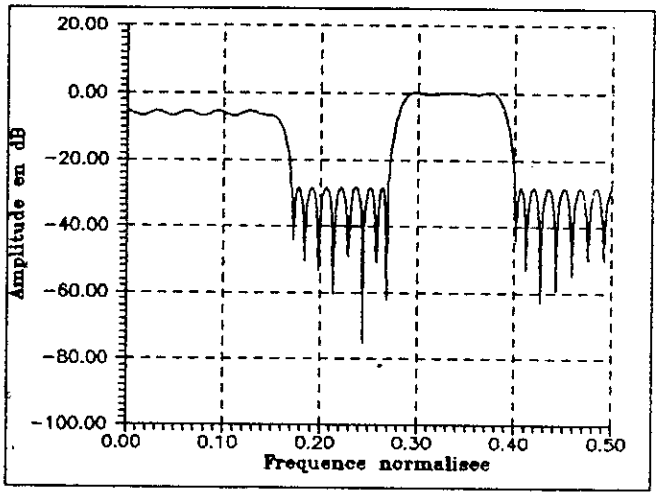


Fig.III-8-Spectre d'amplitude d'un filtre numerique a 4 bandes (exemple III-7) d'ordre 65.

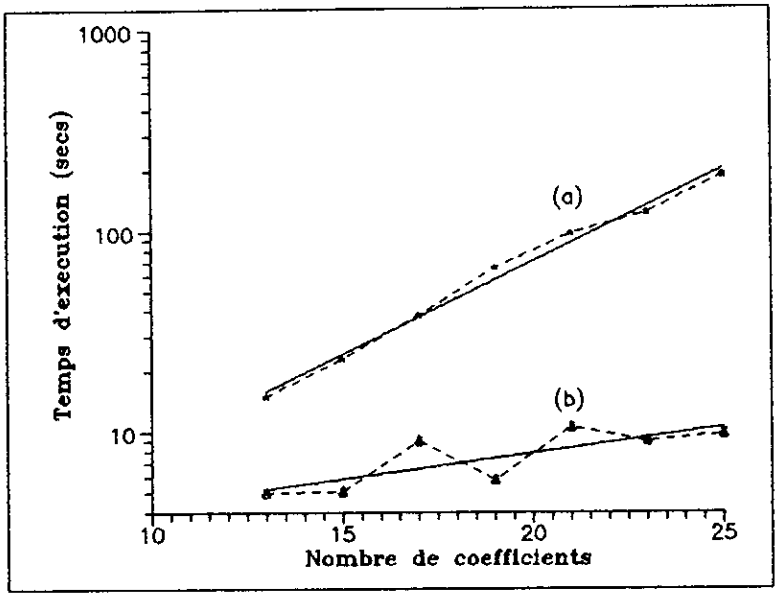


Fig.III-9-Temps d'execution en fonction de N pour un filtre passe-bas
(les courbes d'interpolation sont representees en trait plein)
(a)-Ancienne methode du simplexe.
(b)-Methode du simplexe amelioree.

CHAPITRE IV

OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE SOUMIS A DES CONTRAINTES SIMULTANEES SUR LES CARACTERISTIQUES FREQUENTIELLE ET TEMPORELLE

IV-1-INTRODUCTION

Les contraintes sur la réponse fréquentielle d'un filtre numérique présentées sous forme d'un gabarit fréquentiel ne satisfont pas parfois l'utilisateur. Il peut exiger la réduction - voire l'élimination - des interférences intersymbôles dans un système de télécommunication par exemple, comme il peut exiger la limitation des oscillations indésirables de la réponse indicielle.

Nous allons présenter dans ce chapitre sur un exemple (limitation des oscillations de la réponse indicielle d'un filtre passe-bas) comment on peut associer des contraintes temporelles à des contraintes fréquentielles en utilisant l'approche décrite dans le chapitre précédent (méthode du simplexe améliorée). L'extension à d'autres exemples de contraintes se fait de la même manière à condition que ces dernières se présentent sous forme d'équations ou d'inéquations linéaires en fonction des coefficients du filtre.

IV-2-ANALYSE DU REGIME TRANSITOIRE DE LA REPONSE INDICIELLE D'UN FILTRE NUMERIQUE PASSE-BAS

La réponse indicielle $\{a_k\}$ d'un filtre numérique est la réponse de celui-ci lorsqu'on applique à son entrée un échelon de Heaviside e(i) défini par:

$$e(i) = \begin{cases} 0 & \text{si } i < 0 \\ 1 & \text{si } i \geq 0 \end{cases} \qquad \text{(IV-1)}$$

on a alors

$$a_k = \sum_{i=0}^{k} h(i) \qquad k=0,1,\ldots \qquad \text{(IV-2)}$$

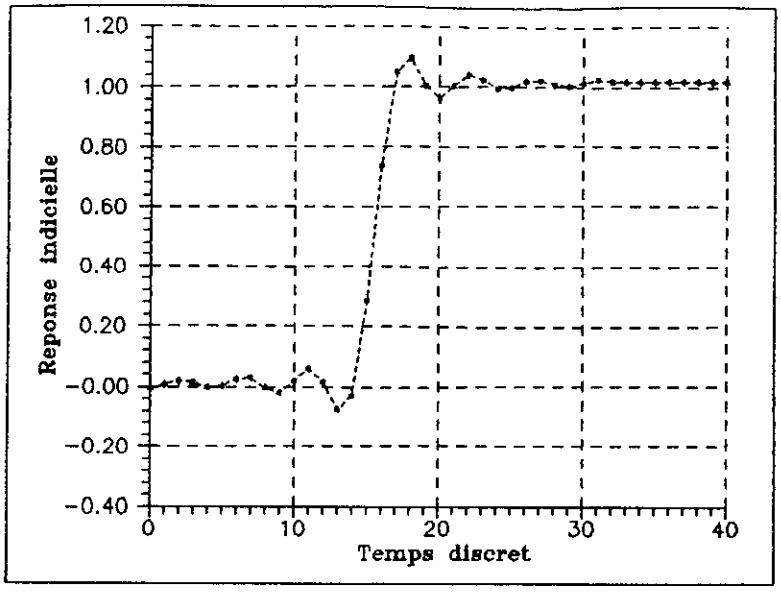


Fig.IV-1-Reponse indicielle d'un filtre numerique RIF du type passe-bas d'ordre 33:

Pour un filtre passe-bas, cette réponse (Fig.IV-1) présente des ondulations indésirables autour de la valeur initiale a_{ini} (voisine de zéro) et autour de la valeur finale a_{fin} (voisine de 1). Ces oscillations doivent être limitées à une valeur maximale Δ spécifiée par l'utilisateur. Nous devons avoir alors:

$$|a_k - a_{ini}| \leq \Delta \quad \text{autour de la valeur initiale} \qquad \text{(IV-3-a)}$$

et

$$|a_k - a_{fin}| \leq \Delta \quad \text{autour de la valeur finale} \qquad \text{(IV-3-b)}$$

IV-3-LIMITATION DES OSCILLATIONS DE LA REPONSE INDICIELLE D'UN FILTRE NUMERIQUE RIF A PHASE LINEAIRE DU TYPE PASSE-BAS

IV-3-1-Elaboration d'un programme

Pour un filtre numérique RIF à phase linéaire du type passe-bas et d'ordre N = 2n+1 on doit avoir:

$$\left| \sum_{i=0}^{k} h(i) \right| \leq \Delta \qquad \text{pour } k=0,1,\ldots,n-2 \qquad \text{(IV-4-a)}$$

$$\left| \sum_{i=0}^{k} h(i) - a_{fin} \right| \leq \Delta \qquad \text{pour } k=n+1,n+2,\ldots,N-2 \qquad \text{(IV-4-b)}$$

Une propriété intéressante caractérisant ce type de filtres est donnée par la relation suivante:

$$|a_o - a_k| = |a_{fin} - a_{N-k-2}|, \text{ pour } k=0,1,\ldots,n-2 \qquad \text{(IV-5)}$$

où

$$a_{fin} = \sum_{i=0}^{N-2} h(i) \qquad \text{(IV-6)}$$

est la valeur finale de la réponse indicielle et a_o sa valeur initiale.

La démonstration de cette relation est très aisée : il suffit d'utiliser la relation (I-15) qui caractérise la linéarité de la phase. En effet, nous pouvons écrire

$$a_{fin} - a_{N-k-2} = \sum_{i=0}^{N-2} h(i) - \sum_{i=0}^{N-k-2} h(i)$$

$$a_{fin} - a_{N-k-2} = \sum_{i=0}^{k} h(i) + \sum_{i=k+1}^{N-2} h(i) - \left(\sum_{i=1}^{N-k-2} h(i) + h(0) \right)$$

et d'après (I-15-a) on a

$$\sum_{i=1}^{N-k-2} h(i) = \sum_{i=k+1}^{N-2} h(i)$$

il vient alors

$$a_{fin} - a_{N-k-2} = \sum_{i=0}^{k} h(i) + \sum_{i=k+1}^{N-2} h(i) - \sum_{i=k+1}^{N-2} h(i) - a_o$$

d'où

$$|a_{fin} - a_{N-k-2}| = |a_o - a_k|$$

Cette propriété est très intéressante, car, si on arrive à minimiser les oscillations autour de zéro, pour i=0 jusqu'à n-2, les oscillations autour de la valeur finale, pour k=n+1 jusqu'à N-1, se trouvent automatiquement limitées à la même borne. En conséquence, il n'est pas nécessaire de limiter les oscillations de la réponse indicielle autour de zéro et de 1, il suffit d'examiner les ondulations autour de zéro seulement. Et cela nous permet de réduire le nombre de contraintes.

Les inégalités (IV-4-a) et (IV-4-b) se présentent comme des combinaisons linéaires des coefficients du filtre, ce qui nous permet de les ajouter aux contraintes du problème de programmation linéaire formulé dans (II-17).

Pour pouvoir adapter ces contraintes supplémentaires au problème principal, décrit au paragraphe II-4, il faut tout d'abord écrire ces mêmes contraintes en fonction des variables d_i conformément au tableau II-2, et cela, pour pouvoir utiliser la forme générale (II-4). Pour le cas considéré ici (N impair et réponse impulsionnelle symétrique) nous avons (voir tableau II-2):

$$\begin{cases} h(i) = 0.5d_{n-i} & \text{pour } i=0,1,\ldots,n-2 \\ h(n) = d_o \end{cases}$$

La relation (IV-4-a) devient alors:

$$\left| 0.5 \sum_{i=n-k}^{n} d_i \right| \leq \Delta \qquad \text{pour } k=0,1,\ldots,n-2 \qquad \text{(IV-7)}$$

puis en fonction des variables x_i, conformément à la relation (II-21):

$$\left| 0.5 \sum_{i=n-k}^{n} (x_i - a) \right| \leq \Delta \qquad \text{pour } k=0,1,\ldots,n-2 \qquad \text{(IV-8)}$$

Soient la matrice A_1 et le vecteur colonne b_1 définis, respectivement, par les relations (IV-9) et (IV-10).

$$A_1 = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0.5 & 0 \\ 0 & 0 & 0 & \ldots & 0.5 & 0.5 & 0 \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot \\ 0 & 0 & 0.5 & \ldots & 0.5 & 0.5 & 0 \\ 0 & +0.5 & +0.5 & \ldots & +0.5 & +0.5 & 0 \\ 0 & 0 & 0 & \ldots & 0 & -0.5 & 0 \\ 0 & 0 & 0 & \ldots & -0.5 & -0.5 & 0 \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot \\ 0 & 0 & 0.5 & \ldots & -0.5 & -0.5 & 0 \\ 0 & -0.5 & -0.5 & \ldots & -0.5 & -0.5 & 0 \end{bmatrix} \qquad \text{(IV-9)}$$

$$b_1=\begin{bmatrix} \Delta + 0.5\times a \\ \Delta + 2\times 0.5\times a \\ \vdots \\ \Delta+(n-1)\times 0.5\times a \\ \Delta + n\times 0.5\times a \\ -\Delta - 0.5\times a \\ -\Delta - 2\times 0.5\times a \\ \vdots \\ -\Delta-(n-1)\times 0.5\times a \\ -\Delta - 0.5\times n\times a \end{bmatrix} \qquad \text{(IV-10)}$$

Compte tenu de ceci, nous pouvons écrire les contraintes formulées dans (IV-8) sous la forme:

$$\begin{cases} A_1 x \le b_1, & x=(x_0,x_1,\ldots,x_n,\varepsilon)^t \\ x_i \ge 0 \end{cases} \qquad \text{(IV-11)}$$

Ces contraintes seront introduites directement dans le problème initial (c-à-d le problème sans contraintes supplémentaires) formulé dans (III-1). Nous pouvons écrire alors:

$$\left[\begin{array}{c|ccccc} & 1 & 0 & 0 & \ldots & 0 \\ A & 0 & 1 & 0 & \ldots & 0 \\ & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdots\cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ & \cdot & \cdot & \cdot & \cdots & \cdot \\ A_1 & \cdot & \cdot & \cdot & \cdots & \cdot \\ & \cdot & \cdot & \cdot & \cdots & \cdot \\ & 0 & 0 & 0 & \ldots & 1 \end{array}\right] .x=\begin{bmatrix} b \\ b_1 \end{bmatrix} \qquad \text{(IV-12)}$$

Ceci est valable pour le cas d'un filtre d'ordre impair et de symétrie positive. Pour les autres cas, ils convient d'utiliser les relations du tableau II-2.

Pour résoudre le problème (IV-12), nous avons utilisé l'algorithme dual-simplexe comme pour l'optimisation sans contraintes supplémentaires car la condition (III-9), nécessaire pour son application, demeure toujours vérifiée; la configuration du tab-

leau simplexe reste pratiquement la même (voir détails au paragraphe III-3-1).

Pour cette raison, nous avons employé le même programme utilisé pour l'optimisation sans contraintes supplémentaires, c'est-à-dire le programme FASTSIMPLEX, et la limitation des oscillations de la réponse indicielle y est considérée comme une option supplémentaire.

IV-3-2-Résultats et commentaires

La méthode a été testée sur plusieurs exemples. Prenons un cas typique, soit le filtre défini par les fonctions suivantes:

$$D(f)=\begin{cases} 1 & f \in [0.00,0.20] \\ 0 & f \in [0.25,0.50] \end{cases} \quad \text{et} \quad W(f)=\begin{cases} 1 & f \in [0.00,0.20] \\ 10 & f \in [0.25,0.50] \end{cases}$$

Le filtre numérique optimal au sens de la norme de Chebyshev a une réponse indicielle dégradée : elle présente des oscillations autour de 0 et de 1 dont l'amplitude est importante surtout lorsque le nombre de coefficients N est faible.

Prenons les cas particuliers suivants:

-*Exemple IV-1*-Filtre défini précédemment avec N=17.
-*Exemple IV-2*-Filtre défini précédemment avec N=25.
-*Exemple IV-3*-Filtre défini précédemment avec N=31.

Les figures IV-2 à IV-4 représentent pour chaque exemple : le spectre d'amplitude du filtre numérique avant et après la limitation des oscillations transitoires de sa réponse indicielle.

Nous pouvons remarquer que, grâce à l'introduction de l'option relative à l'optimisation avec contraintes supplémentaires, la réponse indicielle est bien améliorée et ses oscillations transitoires sont bien limitées à Δ qui vaut ici 0,06 (Tableau IV-1).

Pour mesurer l'efficacité de cette option, nous avons introduit une quantité (excursion maximale) qui mesure l'écart maximal (en valeur absolue) entre la réponse indicielle et sa valeur initiale. Ceci peut être exprimé comme suit:

$$\delta_{max} = \max_{k=0,\ldots,n-2} |a_k - a_{ini}| \qquad \text{(IV-13)}$$

Le tableau IV-1 donne, pour les exemples IV-1 à IV-3, les valeurs de δ_{max} pour:

(a)-Le cas de l'optimisation minimax sans limitation de la réponse indicielle.

(b)-Le cas de l'optimisation minimax avec limitation de la réponse indicielle.

D'après ces résultats, nous constatons que cet écart est nettement amélioré. Par exemple pour N=17, cet écart passe de 0.19 à 0.06.

Tableau IV-1-Valeurs de δ_{max} :
(a)-Optimisation sans limitation de la réponse indicielle.
(b)-Optimisation avec limitation de la réponse indicielle.

	(a)	(b)
Exemple IV-1	0.19	0.06
Exemple IV-2	0.11	0.06
Exemple IV-3	0.13	0.06

Une remarque très importante est à considérer ici : contrairement à ce qu'on s'attendait, le spectre d'amplitude reste pratiquement inchangé (avec ou sans limitation de la réponse indicielle) comme nous pouvons le voir dans les courbes des figures IV-5 à IV-7 (les spectres d'amplitude sont pratiquement superposés). Ce résultat est d'une importance capitale, car nous venons de voir qu'il est possible d'améliorer considérablement la réponse indicielle d'un filtre numérique sans affecter sérieusement son spectre d'amplitude. Ce résultat a été vérifié sur un bon nombre d'exemples.

Le temps d'exécution de la nouvelle procédure du simplexe pour un problème soumis à des contraintes supplémentaires (sur la réponse indicielle), pour le même nombre de coefficients, est supérieur au temps d'exécution de cette même procédure sans contraintes supplémentaires (Fig.IV-8), mais, il reste toujours largement inférieur à celui de l'ancienne méthode du simplexe. L'augmentation du temps d'exécution est dû, bien sûr, à l'augmentation du rang du tableau simplexe de 2×n lignes.

IV-4-CONLUSION

Nous avons présenté ici un seul type de contraintes supplémentaires mais d'une importance certaine. L'utilisateur peut introduire d'autres contraintes supplémentaires se présentant sous forme d'équations ou d'inéquations linéaires en fonction des coefficients du filtre. La méthode est la même et l'utilisateur n'a qu'à suivre les étapes décrites plus haut (voir § IV-3). En fin, nous sommes arrivés à un résultat très important, c'est que la minimisation des ondulations de la réponse indicielle d'un filtre numérique RIF à phase linéaire n'affecte pas sérieusement la réponse fréquentielle (et en particulier le spectre d'amplitude) de ce dernier : un résultat qui a été vérifié sur la plupart des exemples traités dans cette thèse.

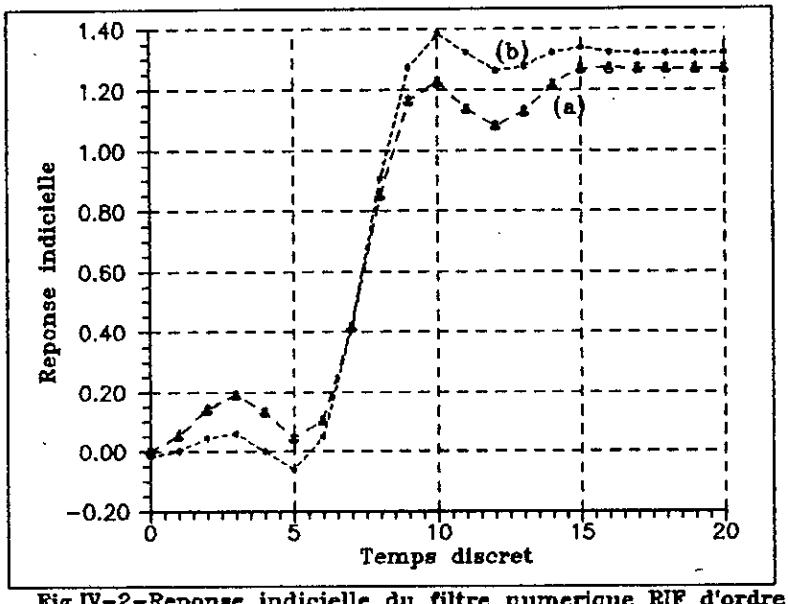


Fig.IV-2-Reponse indicielle du filtre numerique RIF d'ordre 17 defini par l'exemple IV-1.'
(a)-sans limitation de la reponse indicielle.
(b)-avec limitation de la reponse indicielle.

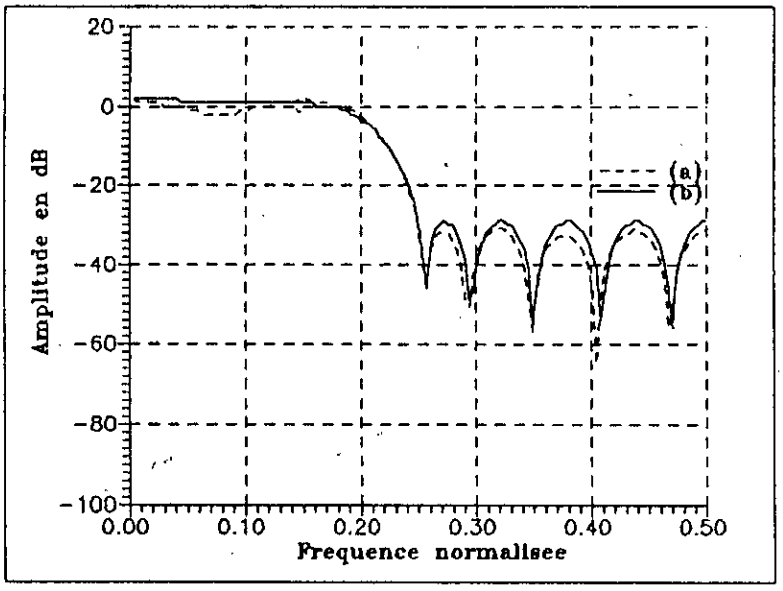


Fig. IV-5-Spectre d'amplitude du filtre numerique defini par l'exemple IV-1.
(a)-Sans limitation de la reponse indicielle.
(b)-Avec limitation de la reponse indicielle.

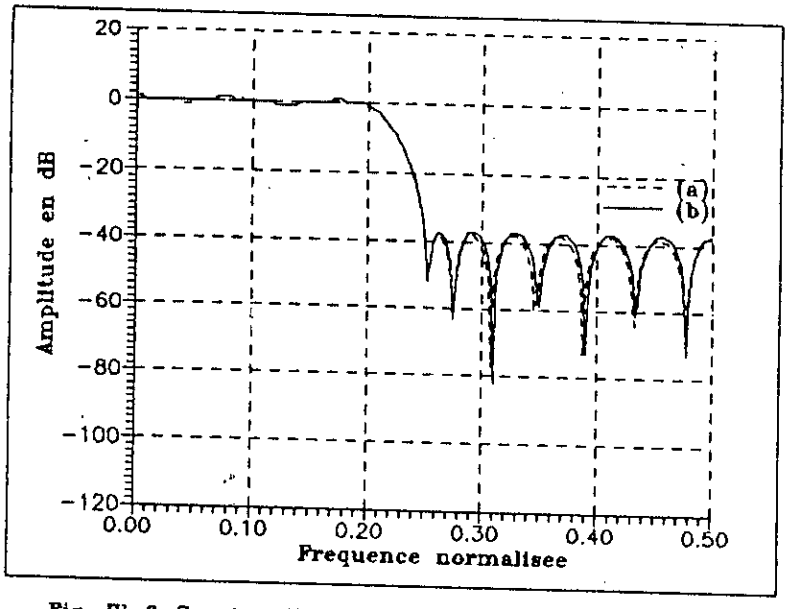


Fig. IV-6-Spectre d'amplitude du filtre numerique defini par l'exemple IV-2.
(a)-Sans limitation de la reponse indicielle.
(b)-Avec limitation de la reponse indicielle.

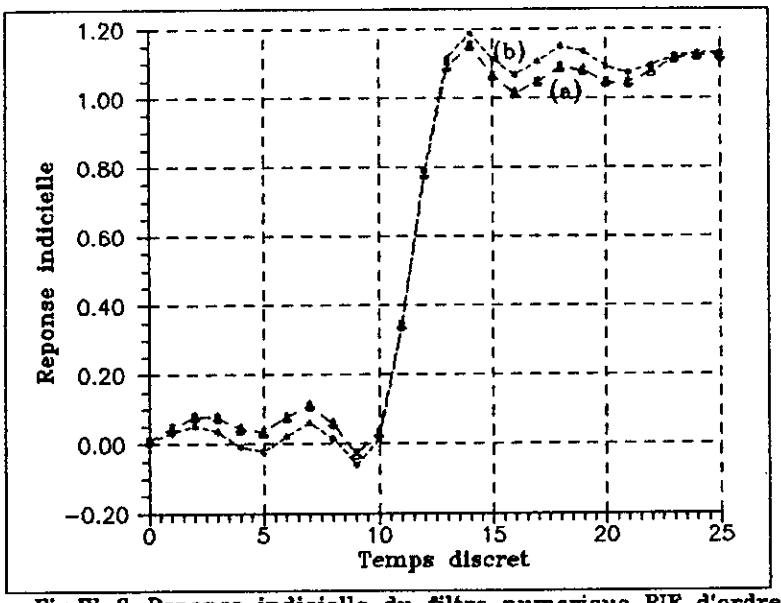


Fig.IV-3-Reponse indicielle du filtre numerique RIF d'ordre 25 defini par l'exemple IV-2.
(a)-sans limitation de la reponse indicielle.
(b)-avec limitation de la reponse indicielle.

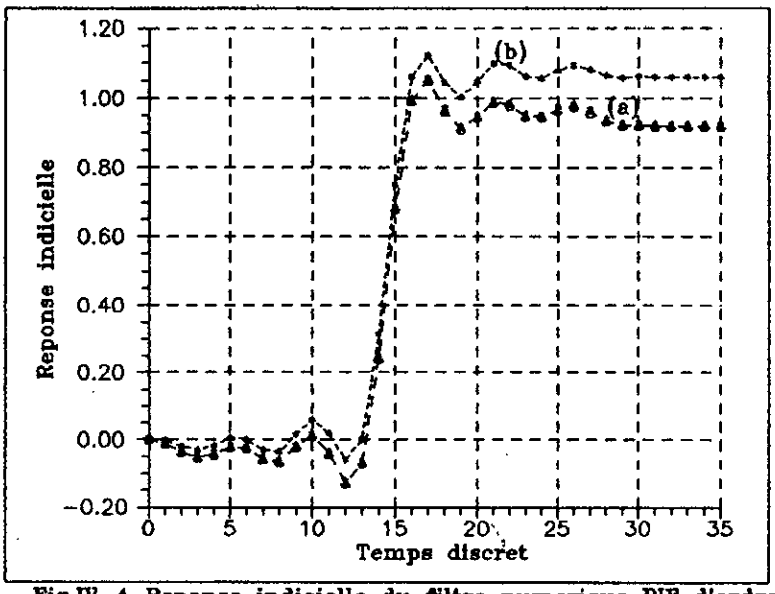


Fig.IV-4-Reponse indicielle du filtre numerique RIF d'ordre 31 defini par l'exemple IV-3.
(a)-sans limitation de la reponse indicielle.
(b)-avec limitation de la reponse indicielle.

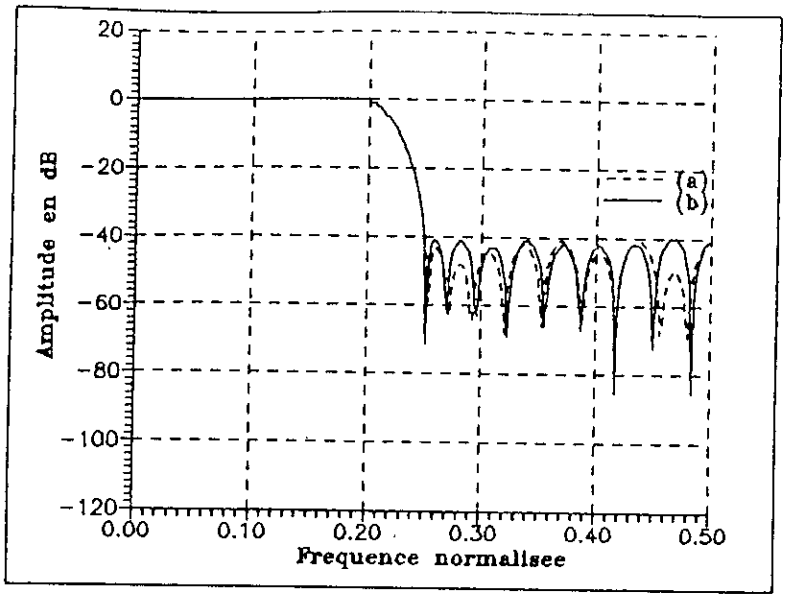


Fig. IV-7-Spectre d'amplitude du filtre numerique defini par l'exemple IV-3.
(a)-Sans limitation de la reponse indicielle.
(b)-Avec limitation de la reponse indicielle.

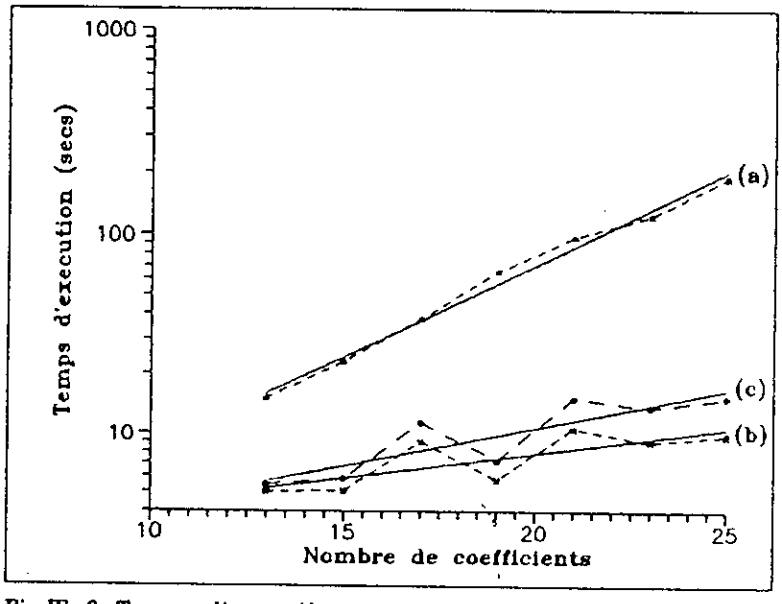


Fig.IV-8-Temps d'execution en fonction de N pour un filtre passe-bas (les courbes d'interpolation sont representees en ligne continue):
(a)-Ancienne methode du simplexe sans limitation de la rep. indic.
(b)-Nouvelle methode du simplexe sans limitation de la rep. indic.
(c)-Nouvelle methode du simplexe avec limitation de la rep. indic.

CHAPITRE V

OPTIMISATION DES FILTRES NUMERIQUES RIF A PHASE LINEAIRE ET COEFFICIENTS DE LONGUEUR FINIE PAR UNE METHODE DE RECHERCHE LOCALE

V-1-INTRODUCTION

Les méthodes d'optimisation des filtres numériques RIF, proposées dans les chapitres précédents, supposent que les coefficients sont de longueur infinie. Or, dans une réalisation pratique, la précision est finie et toutes les grandeurs sont codées sur un nombre de bits limité. Le nombre de bits alloué aux coefficients est donc fini. Leur troncature entraîne, par la suite, une modification de la réponse fréquentielle du filtre se traduisant par une dégradation de l'erreur maximale optimale. Pour y remédier, on est obligé d'ajuster ces coefficients à des valeurs qui améliorent au mieux l'erreur maximale.

Les méthodes optimales utilisées pour résoudre ce genre de problèmes sont généralement basées sur des techniques classiques de l'optimisation en nombres entiers qui demandent un temps de calcul important sur ordinateur. Nous allons présenter dans ce chapitre une méthode approchée (sub-optimale) qui consiste à manipuler les coefficients quantifiés, en examinant leur voisinage immédiat, dans le but d'obtenir la combinaison qui donne le minimum du maximum de la norme de l'erreur moyennant un temps d'exécution raisonnable.

V-2-INFLUENCE DE LA LIMITATION DU NOMBRE DE BITS SUR LA NORME DE L'ERREUR MAXIMALE

Soient h(0),h(1),...,h(N-1) les N coefficients d'un filtre numérique RIF obtenus par l'optimisation en précision infinie (codés sur un nombre de bits illimité) et $\mathcal{E}_{inf}$ l'erreur maximale optimale correspondante (au sens de Chebyshev). Soit b le nombre de bits alloué à chaque coefficient dans une certaine réalisation.

Au cours des opérations arithmétiques, les coefficients subissent une troncature et les valeurs quantifiées par arrondi sont données par la relation

$$h'(i)=2^{-(b-1)}.[h(i).2^{b-1}] \qquad (V-1)$$

où les crochets dénotent la valeur arrondie la plus proche du nombre qui se trouve entre eux. Notons que, sur les b bits alloués aux coefficients, un bit est réservé au signe. De plus, les coefficients arrondis peuvent être représentés simplement par des nombres entiers car la quantité $2^{-(b-1)}$ est un facteur commun indépendant de chaque coefficient.

La nouvelle réponse fréquentielle est donnée par la relation:

$$H'(f)=\sum_{i=0}^{N-1} h'(i)e^{-j2\pi if} \qquad (V-2)$$

La troncature des coefficients entraîne une erreur maximale ε_{arr} qui ne peut être que supérieure ou égale à ε_{inf} (car ε_{inf} correspond à la meilleure approximation).

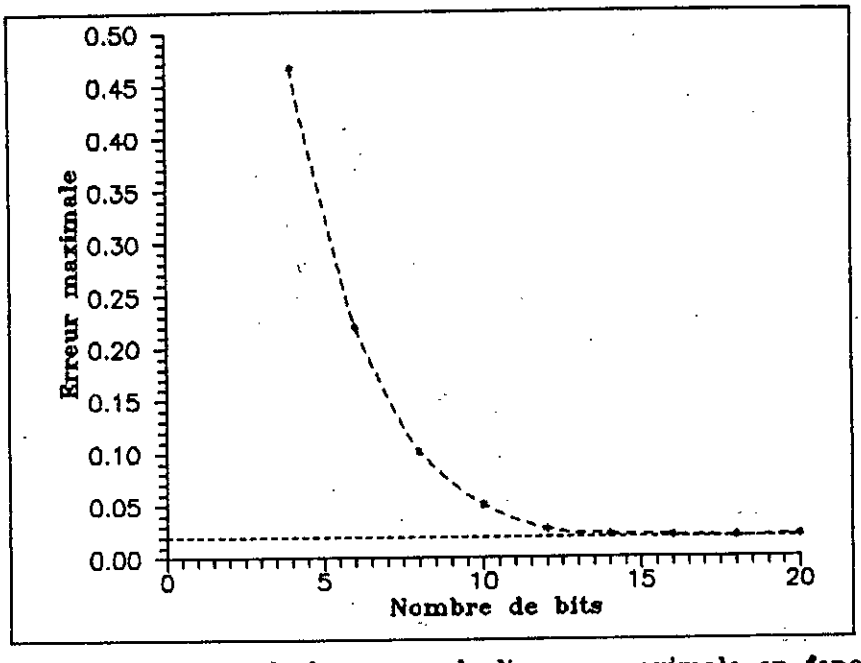


Fig.V-1-Evolution de la norme de l'erreur maximale en fonction du nombre de bits lors d'une quantification par arrondi pour un filtre numerique passe-bas d'ordre 33.

La courbe de la figure V-1 donne l'évolution de cette erreur (c'est-à-dire ε_{arr}) en fonction de b pour un filtre numérique RIF du type passe-bas d'ordre 33 (exemple III-1). Nous remarquons que pour les valeurs faibles de b l'écart entre ε_{arr} et ε_{inf} est important, par exemple, pour b=8, on a ε_{arr}=0.1, elle est 5 fois plus grande que ε_{inf} (qui est égale à .0214). Tandis que pour les valeurs élevées de b, l'écart tend vers zéro. Dans l'exemple cité

plus haut, à partir de b=16, l'écart relatif entre ε_{arr} et ε_{inf} devient inférieur à 1%. Cette analyse quantitative illustrée sur cet exemple particulier est, du point de vue qualitatif, générale et le problème de la troncature se pose chaque fois qu'on a affaire à des valeurs faibles de b.

V-3-METHODE DE LA RECHERCHE LOCALE

V-3-1-Position du problème

La question qu'on peut se poser, c'est comment réduire l'influence de la troncature pour un nombre de bits donné? La réponse à cette question nous conduit à deux types de méthodes:

i) Le premier type de méthodes consiste à déterminer les coefficients h(i) du filtre numérique en imposant au départ leur domaine d'éxistence, soit l'ensemble

$$\{-1,\ldots,-2^{-(b-1)},0,2^{-(b-1)},\ldots,1\}$$

ceci équivaut, bien sûr, à déterminer des coefficients entiers appartenant au domaine

$$\{-2^{b-1},\ldots,-1,0,1,\ldots,2^{b-1}\}$$

En d'autres termes, la condition que les coefficients soient entiers est introduite dans le problème initial sous forme de contraintes supplémentaires qui viennent s'ajouter aux contraintes fréquentielles. La solution obtenue par ce type de méthodes est optimale globalement et une technique qui repose sur l'algorithme du simplexe existe [26,27,28] (méthode de Gomory). Cependant, la covergence de cette dernière est trop lente : le temps d'exécution nécessaire pour résoudre un problème avec un nombre de coefficients moyen est loin d'être raisonnable [11,22].

ii) Le second type de méthodes repose sur l'analyse combinatoire. Elle consiste à résoudre le problème en précision infinie comme première étape. Puis, en partant de la solution arrondie, de générer toutes les combinaisons possibles de jeux de coefficients et d'en extraire celle qui donne le minimum du maximum de l'erreur [11,29]. Une solution intuitive se présente au premier plan pour ce type de méthodes; il s'agit de parcourir toutes les combinaisons qui puissent se présenter (du fait que les coefficients ne peuvent prendre que des valeurs quantifiées bien

déterminées) et d'en extraire celle qui donne le meilleur résultat. Cette solution garantit toujours l'obtention de l'optimum global, mais il est inconcevable de la mettre en oeuvre car le nombre de combinaisons possibles pour un filtre d'ordre moyen est extrêmement grand.

Nous allons partir de cette recherche glogale pour résoudre ce problème en procédant par une *recherche locale* [11,31] afin de réduire au maximum le nombre de combinaisons. La solution obtenue n'est pas forcément optimale globalement. On parle, dans ce cas, de solution sub-optimale.

La méthode de la recherche locale repose sur une remarque intuitive très importante, c'est que la solution recherchée (c'est-à-dire, la solution optimale) ne s'écarte pas beaucoup de la solution arrondie (c'est-à-dire, le jeu de coefficients arrondis). Il suffit donc d'examiner le voisinage immédiat de cette dernière pour aboutir à une solution satisfaisante. Par ailleurs, pour atteindre la solution optimale, il suffit de perturber quelques coefficients seulement de la solution arrondie.

Soit L le nombre maximum de coefficients quantifiés qui peuvent subir un changement lorsqu'on passe de la solution arrondie à la solution optimale. On définit pour chaque coefficient h'_i un voisinage d'ordre M par:

$$V_M = \{ h''_i \text{ tel que } h''_i = h'_i + m.2^{-(b-1)} \quad -M \le m \le M \} \qquad (V-3)$$

Le nombre de solutions à examiner par la recherche locale est donc donné par les paramètres L et M. Soit V_L^M l'ensemble de ces solutions réalisables.

Pour fixer les idées, prenons le cas le plus simple qui puisse se présenter devant nous; le cas où un seul coefficient peut changer à la fois (L=1) avec une perturbation d'une unité positive ou négative (M=1). Les combinaisons possibles sont les suivantes:

$$V_1^1 = \{(h'_0 \pm 1, h'_1, \ldots, h'_{N-1}), \ldots, (h'_0, h'_1, \ldots h'_{i-1}, h'_i \pm 1, h'_{i+1}, \ldots, h'_{N-1}),$$
$$\ldots, (h'_0, h'_1, \ldots, h'_{N-1} \pm 1)\}$$

le nombre de ces combinaisons est 2N.

Nous pouvons citer aussi le cas où deux coefficients au plus (L=2) peuvent subir une perturbation d'une unité positive ou négative (M=1). Dans ce cas les combinaisons possibles sont les

suivantes:

$$\mathcal{V}_2^1=\{(h_0'\pm1,h_1',h_2',\ldots,h_i',\ldots,h_j',\ldots,h_{N-1}'),\ldots,$$
$$(h_0',h_1',\ldots,h_{i-1}',h_i'\pm1,h_{i+1}',\ldots,h_{j-1}',h_j'\pm1,h_{j+1}',\ldots,h_{N-1}'),\ldots,$$
$$(h_0',h_1',h_2',\ldots,h_i',\ldots,h_j',\ldots,h_{N-2}'\pm1,h_{N-1}'\pm1)\}.$$

et leur nombre est égal à $2N^2$.

Pour L=3 et M=1, nous pouvons montrer que le nombre de combinaisons est proportionnel à N^3

V-3-2-Procédure de la recherche locale

Le principe de la méthode de la recherche locale (Fig.V-2) est très simple. A partir de la solution arrondie s donnée par

$$s=\{h'(i)=2^{-(b-1)}.[h(i).2^{b-1}],\ i=0,\ldots,N-1\} \qquad (V-4)$$

(où les crochets dénotent la valeur arrondie et b est le nombre de bits), on cherche s'il existe une solution $s'\in \mathcal{V}_L^M$ telle que:

$$\mathcal{E}(s') < \mathcal{E}(s) \qquad (V-5)$$

où $\mathcal{E}$ désigne la norme de l'erreur maximale au sens de Chebyshev. Si tel est le cas on remplace s par s' et on recommence pour une nouvelle solution appartenant à $\mathcal{V}_L^M$. Si toutes les solutions sont examinées, le minimum local (c'est-à-dire la solution sub-optimale) est obtenu. Le temps d'exécution de cette recherche dépend directement du cardinal de $\mathcal{V}_L^M$.

La méthode de la recherche locale est décrite par l'organigramme de la figure V-2. Pour la génération des combinaisons, nous avons utilisé des boucles imbriquées s'étendant sur toutes les valeurs des indices et qui sont fixés par L et M.

Pour le calcul de la fonction erreur de la combinaison courante $E_c'(f)$ (Fig.V-2), nous pouvons utiliser la fonction erreur du niveau précédent soit $E_c(f)$. En effet, on a la relation [**11**]:

$$E_c(f) = E_c'(f) + W(f).P(i,j,f) \qquad (V-6)$$

où P(i,j,f) est une fonction de perturbation qui ne dépend que des variations des coefficients entre les deux niveaux. Par exemple pour M=2 et L=1, et pour le premier cas du tableau II-2 (ordre impair, symétrie positive), elle est donnée par [**11**]:

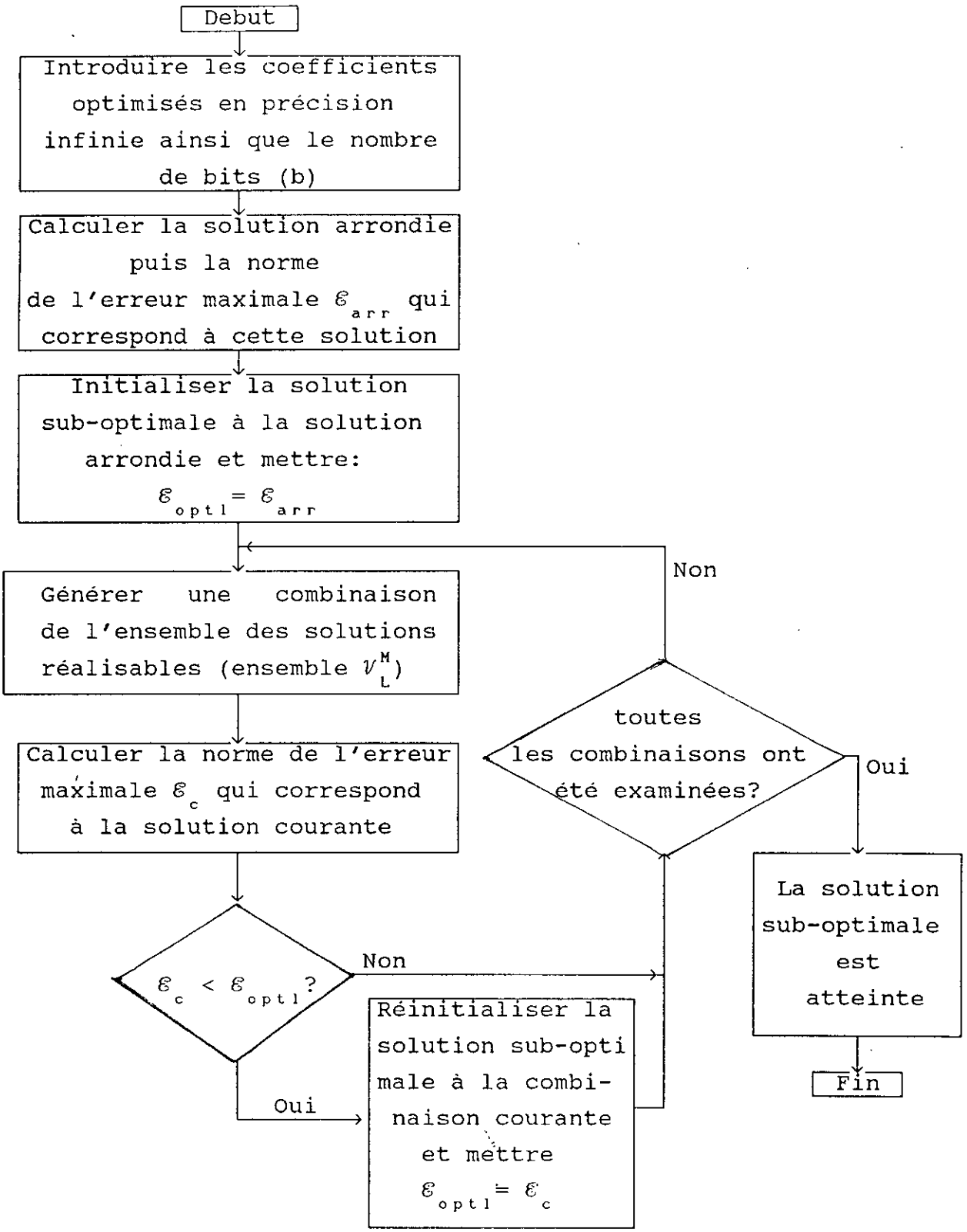


Fig.V-2-Algorithme de la recherche locale.

$$P(i,j,f) = \pm 2^{-(b-1)}.\cos(2\pi if) \pm 2^{-(b-1)}.\cos(2\pi jf) \qquad (V-7)$$

Il est clair que, plus L et M sont grands, plus l'erreur maximale sub-optimale, obtenue par la recherche locale, tend vers l'erreur maximale correspondant à la solution globalement optimale. Cependant, le nombre de combinaisons à examiner par la recherche locale croît rapidement entraînant ainsi un temps de calcul considérable. L'efficacité de la méthode de la recherche locale est mesurée par l'écart entre l'erreur maximale obtenue par cette dernière et l'erreur maximale globalement optimale ainsi que par le temps de calcul nécessaire pour son exécution. Le but visé est, par conséquent, de déterminer les valeurs de L et de M qui assurent toujours le meilleur compromis entre erreur maximale faible et temps d'exécution raisonnable.

V-4-RESULTATS ET COMMENTAIRES

Sur la base des indications du paragraphe précédent, nous avons élaboré un programme de recherche locale (LOCALSEARCH, Annexe II). Le programme est écrit en Fortran77 et la machine utilisée est le VAX/VMS 785. Nous l'avons testé pour plusieurs valeurs de L et M et sur plusieurs types de filtres dont voici quelques exemples:

-*Exemple V-1*: Filtre numérique RIF passe-bas d'ordre N=25 défini par le gabarit de l'exemple III-1. Nombre de bits b=6.

-*Exemple V-2*: Filtre numérique RIF passe-bas d'ordre N=25 défini par le gabarit de l'exemple III-1. Nombre de bits b=8.

-*Exemple V-3*: Filtre numérique RIF passe-bande d'ordre N=21 défini par le gabarit de l'exemple III-2. Nombre de bits b=6.

-*Exemple V-4*: Filtre numérique RIF passe-bande d'ordre N=21 défini par le gabarit de l'exemple III-2. Nombre de bits b=8.

Les tableaux V-1 à V-4 résument les résultats obtenus en donnant, pour différentes valeurs de L et de M, l'erreur maximale et, juste en dessous, le temps de calcul (évalué en secondes) pour chaque exemple cité plus haut.

D'après ces résultats nous constatons que l'erreur maximale globalement optimale en précision finie est atteinte pour des valeurs faibles de L et M. Par ailleurs, le temps d'exécution

croît rapidement en fonction de ces deux paramètres. La meilleure solution est donc celle qui donne le meilleur compromis entre l'erreur maximale et le temps d'execution. La recherche locale qui correspond à L=2 et M=1 nous parait satisfaisante dans la plupart des cas et c'est celle-ci que nous avons choisie pour l'optimisation en nombres entiers.

Tableau V-1-Valeurs de l'erreur maximale et du temps d'exécution pour différentes valeurs de L et de M (exemple V-1).

(ε_{inf}=0.040 ε_{arr}=0.153)

L \ M	1	2	3
1	**0.112** 7	**0.112** 140	**0.112** 480
2	**0.108** 11	**0.085** 280	**0.085** 2350
3	**0.108** 16	**0.085** 2010	**0.085** 5850

Tableau V-2-Valeurs de l'erreur maximale et du temps d'exécution pour différentes valeurs de L et de M (exemple V-2).

(ε_{inf}=0.040, ε_{arr}=0.063)

L \ M	1	2	3
1	**0.055** 7	**0.055** 140	**0.055** 480
2	**0.052** 11	**0.048** 280	**0.048** 2350
3	**0.050** 16	**0.048** 2010	**0.048** 5850

Tableau V-3-Valeurs de l'erreur maximale et du temps d'exécution pour différentes valeurs de L et de M (exemple V-3).

(ε_{inf}=0.023, ε_{arr}=0.098)

L \ M	1	2	3
1	**0.066** 3	**0.066** 42	**0.066** 225
2	**0.036** 5	**0.032** 117	**0.032** 917
3	**0.033** 7	**0.032** 381	**0.032** 1550

Tableau V-4-Valeurs de l'erreur maximale et du temps d'exécution pour différentes valeurs de L et de M (exemple V-4).

(ε_{inf}=0.023, ε_{arr}=0.037)

L \ M	1	2	3
1	**0.025** 3	**0.025** 42	**0.025** 225
2	**0.025** 5	**0.025** 117	**0.025** 917
3	**0.025** 7	**0.025** 381	**0.025** 1550

Les courbes des figures V-3 à V-6 représentent les spectres fréquentiels de l'erreur pour les filtres des exemples précédents optimisés par la recherche locale correspondant à L=2 et M=1. Chaque courbe est comparée au cas où la précision est infinie et au cas de la quantification par arrondi. On trouve, représentées également, les erreurs maximales correspondant à chaque cas.

Le tableau V-5 donne, pour les différents exemples cités plus haut, l'erreur maximale obtenue en précision infinie (ε_{inf}), l'erreur maximale correspondant au jeu de coefficients arrondis (ε_{arr}), l'erreur maximale obtenue par la recherche globale (ε_{optg}), l'erreur maximale obtenue par la recherche locale correspondant à L=2 et M=1, et un écart relatif e_r entre ces deux dernières erreurs défini par [31]:

$$e_r = \frac{\varepsilon_{arr} - \varepsilon_{optl}}{\varepsilon_{arr} - \varepsilon_{optg}}$$

Tableau V-5-Erreurs maximales obtenues pour différents cas d'optimisation:
-Précision infinie.
-Quantification par arrondi.
-Quantification par arrondi avec recherche globale.
-Quantification par arrondi avec recherche locale.

Exemple	ε_{inf}	ε_{arr}	ε_{optg}	ε_{optl}	e_r
Exemple V-1	0.040	0.154	0.085	0.108	*0.67*
Exemple V-2	0.040	0.063	0.048	0.052	*0.73*
Exemple V-3	0.023	0.098	0.032	0.037	*0.92*
Exemple V-4	0.023	0.036	0.025	0.025	*1.00*

Ce rapport est une mesure de l'efficacité de la recherche locale vis-à-vis de l'erreur maximale, plus il est proche de 1, plus l'erreur maximale obtenue par la recherche locale est proche de l'erreur maximale obtenue par la recherche globale.

Nous pouvons affirmer, d'après les résultats obtenus, que la recherche locale correspondant à L=2 et M=1 est satisfaisante dans la plupart des cas ce qui justifie bien nôtre choix.

V-5-INFLUENCE DE LA QUANTIFICATION SUB-OPTIMALE SUR LA REPONSE INDICIELLE

La quantification agit directement sur les coefficients, elle a donc une influence directe sur la réponse indicielle. Essayons de voir sur les exemples IV-1 à IV-3 (cf. § V-3-2), pour lesquels nous avons déjà limité les oscillations de la réponse indicielle, quelle est l'influence de la quantification (par la recherche

locale correspondant à L=2 et M=1) sur la réponse indicielle. Le nombre de bits étant pris faible (égal à 6). Les courbes des figures V-7 à V-9 illustrent, pour chacun de ces trois exemples, la réponse indicielle non limitée, la réponse indicielle améliorée (limitée) et la réponse indicielle améliorée obtenue après quantification sub-optimale.

Le tableau V-6 donne, pour ces trois exemples, l'écart maximal δ_{max} entre la réponse indicielle et sa valeur initiale défini par la relation (IV-13).

D'après ces résultats, nous constatons que cet écart reste relativement faible après quantification sub-optimale ce qui montre que cette dernière a peu d'influence sur la réponse indicielle. Cette règle est générale, nous l'avons vérifiée sur un très grand nombre d'exemples.

Tableau V-6. Valeurs de δ_{max}:
(a)-Optimis. sans limit. de la rép. indicielle ni quantification.
(b)-Optimis. avec limit. de la rép. indicielle et sans quantific.
(c)-Optimis. avec limit. de la rép. indicielle et quantif.sub-opt.

	(a)	(b)	(c)
N = 17	0.19	0.06	0.06
N = 25	0.11	0.06	0.06
N = 31	0.13	0.06	0.08

V-6-CONCLUSION

Nous avons présenté ici une technique d'optimisation en nombres entiers basée sur un examen local du voisinage des coefficients arrondis. L'expérience a montré qu'il suffit de perturber deux coefficients seulement à la fois, au plus d'une unité (positive ou négative), pour obtenir le meilleur compromis entre l'erreur maximale et le temps d'exécution.

Nous avons examiné aussi l'influence de la quantification sur la réponse indicielle optimisée au chapitre IV et nous avons constaté que cette influence est faible.

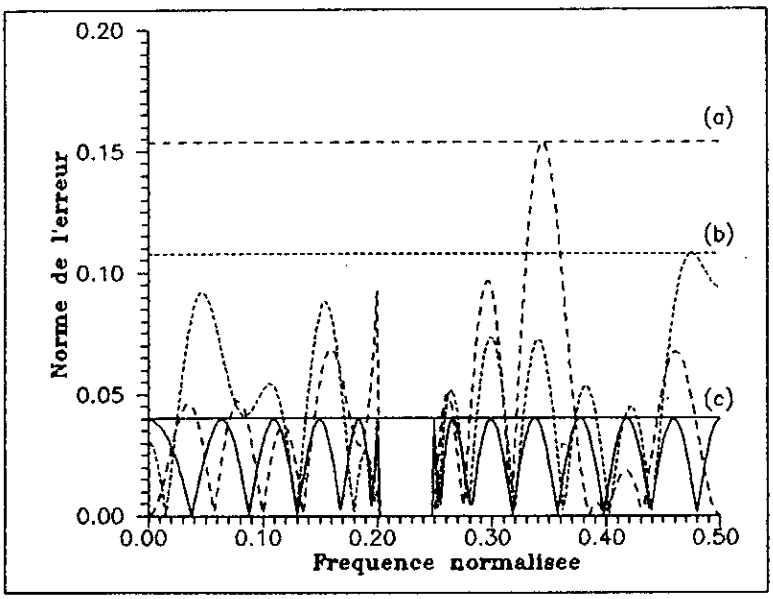


Fig. V-3. Spectre frequentiel de la norme de l'erreur pour un filtre numerique RIF passe-bas d'ordre 25 (exemple V-1)
(a)-Optimis. et quantification par arrondi (b=6).
(b)-Optimis. et quantifi. sub-optimale par recherche locale (L=2,M=1).
(c)-Optimisation sans quantification.

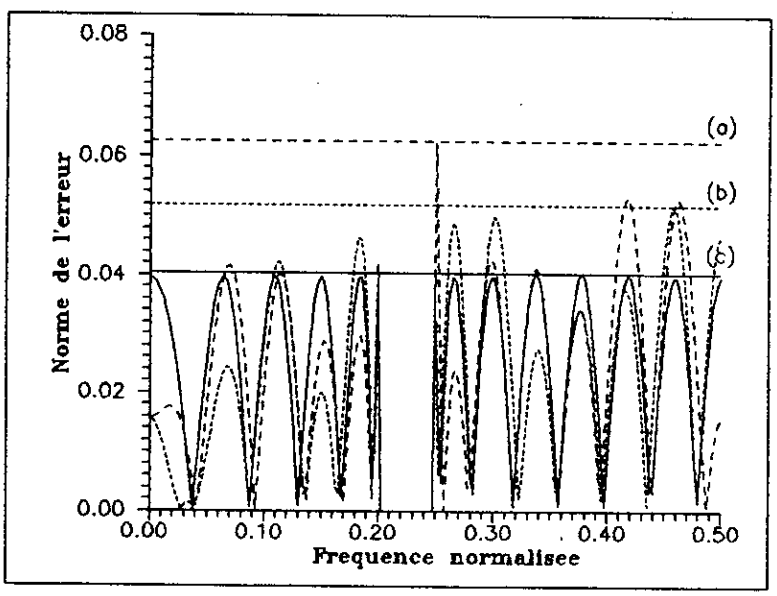


Fig. V-4. Spectre frequentiel de la norme de l'erreur pour un filtre numerique RIF passe-bas d'ordre 25 (exemple V-2)
(a)-Optimis. et quantification par arrondi (b=6).
(b)-Optimis. et quantifi. sub-optimale par recherche locale (L=2,M=1).
(c)-Optimisation sans quantification.

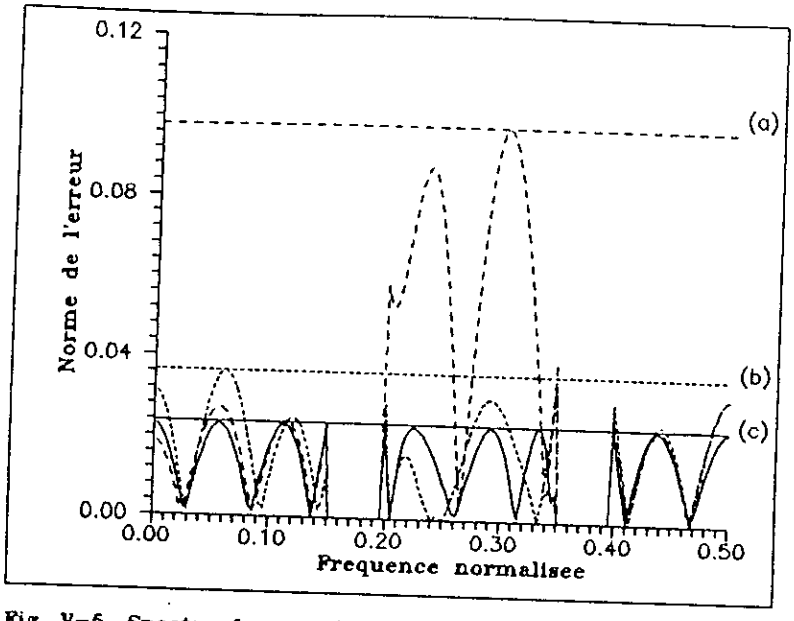


Fig. V-5. Spectre frequentiel de la norme de l'erreur pour un filtre numerique RIF passe-bande d'ordre 21 (exemple V-3)
(a)-Optimis. et quantification par arrondi (b=6).
(b)-Optimis. et quantifi. sub-optimale par recherche locale (L=2,M=1).
(c)-Optimisation sans quantification.

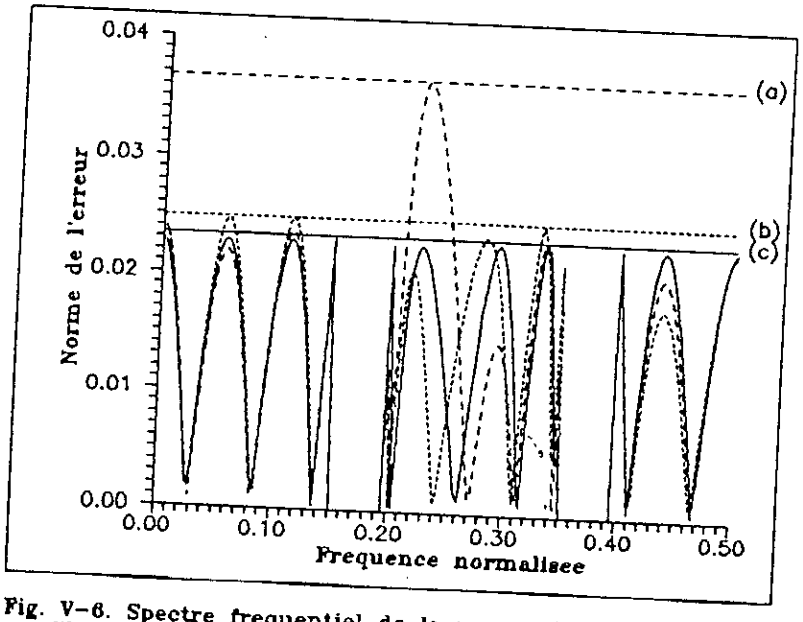


Fig. V-6. Spectre frequentiel de la norme de l'erreur pour un filtre numerique RIF passe-bande d'ordre 21 (exemple V-4)
(a)-Optimis. et quantification par arrondi (b=6).
(b)-Optimis. et quantifi. sub-optimale par recherche locale (L=2,M=1).
(c)-Optimisation sans quantification.

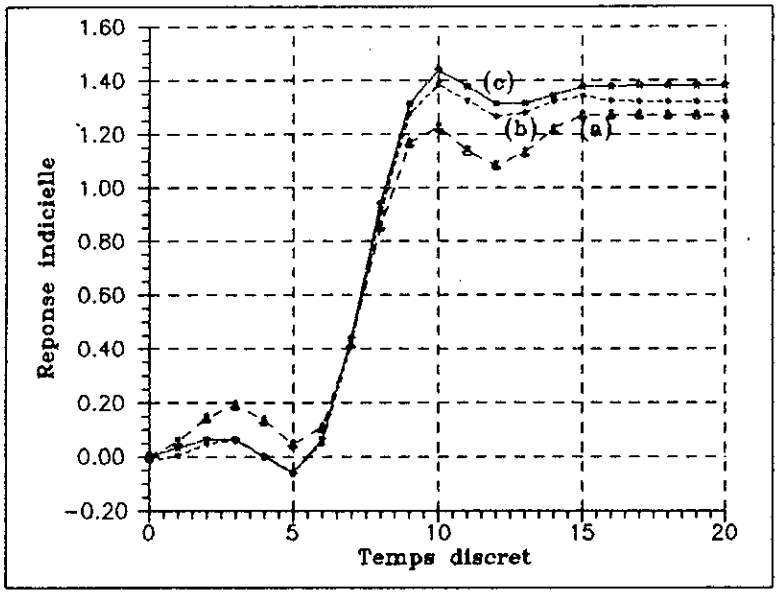


Fig.V-7-Reponse indicielle du filtre numerique RIF de l'exemple IV-1:
(a)-Optimis. sans limitation de la rep. indici. ni quantifi. sub-optimale.
(b)-Optimis. avec limitation de la rep. indici. sans quantifi. sub-optimale.
(c)-Optimis. avec limitation de la rep. indici. et quantifi. sub-optimale.

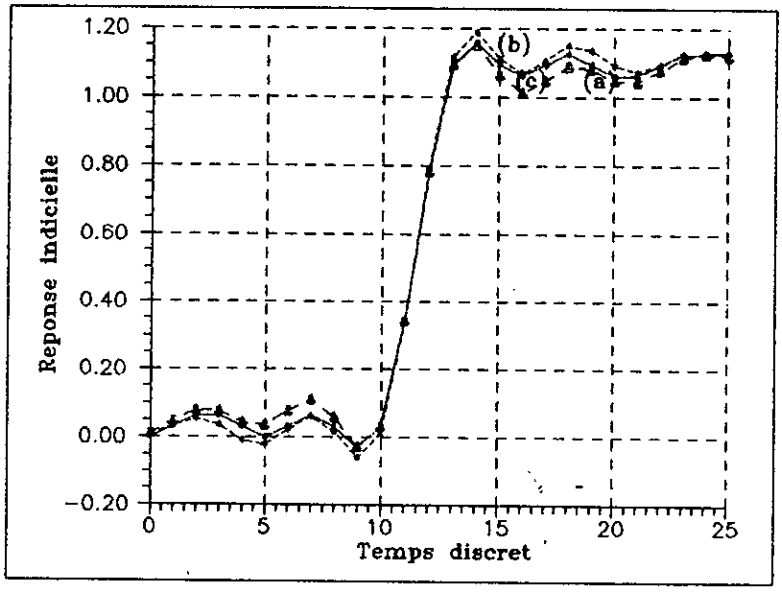


Fig.V-8-Reponse indicielle du filtre numerique RIF de l'exemple IV-2:
(a)-Optimis. sans limitation de la rep. indici. ni quantifi. sub-optimale.
(b)-Optimis. avec limitation de la rep. indici. sans quantifi. sub-optimale.
(c)-Optimis. avec limitation de la rep. indici. et quantifi. sub-optimale.

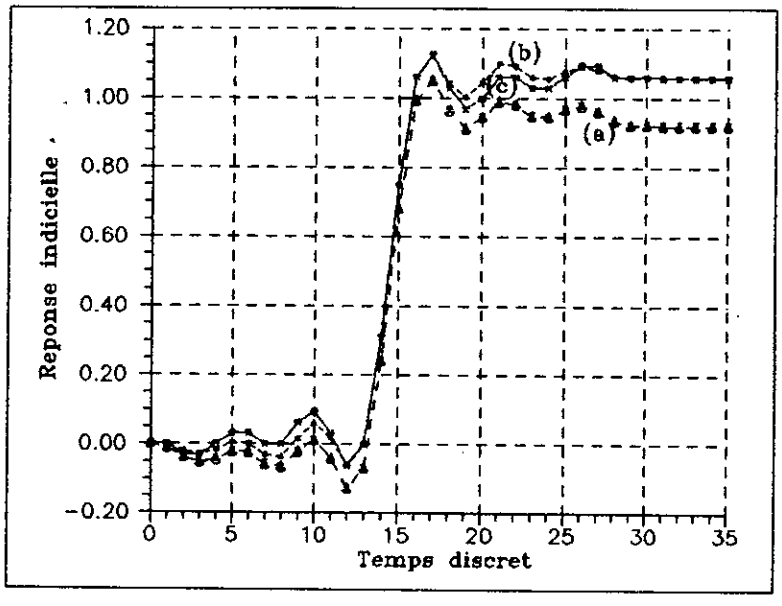


Fig.V-9-Reponse indicielle du filtre numerique RIF de l'exemple IV-3:
(a)-Optimis. sans limitation de la rep. indici. ni quantifi. sub-optimale.
(b)-Optimis. avec limitation de la rep. indici. sans quantifi. sub-optimale.
(c)-Optimis. avec limitation de la rep. indici. et quantifi. sub-optimale.

CONCLUSION

Nous avons présenté, dans cette thèse, une procédure améliorée pour l'optimisation des caractéristiques fréquentielles au sens de Chebyshev des filtres RIF à phase linéaire basée sur l'algorithme du simplexe. Cette approche peut être considérée comme une amélioration de la performance de la méthode classique utilisant la programmation linéaire (algorithme du simplexe). L'avantage principal de cette approche par rapport à la méthode rapide de Remez est le fait d'associer aux contraintes fréquentielles (gabarit fréquentiel) des contraintes se rapportant au domaine temporel.

Nous avons élaboré un programme (que nous avons appelé FASTSIMPLEX) traduisant la méthode du simplexe améliorée. Plusieurs exemples ont été considérés pour tester cette procédure améliorée : les résultats sont identiques à ceux obtenus avec l'ancienne méthode du simplexe avec un temps d'exécution nettement inférieur.

Le cas où le problème présente des contraintes linéaires supplémentaires liées au domaine temporel en plus des contraintes fréquentielles a été aussi évoqué, et les résultats obtenus étaient satisfaisants.

Une méthode d'optimisation des filtres numériques RIF à coefficients de longueur finie a été aussi présentée. Cette méthode, qui repose sur une exploration locale du voisinage immédiat de la solution arrondie, présente le meilleur compromis entre la norme de l'erreur au sens de Chebyshev et le temps d'exécution.

Enfin, nous devons mentionner que ce travail n'est pas destiné à une application restreinte ni représente la solution pratique à un problème particulier, il entre dans le cadre général de l'optimisation des filtres numériques RIF à phase linéaire aux applications diverses. Parmi ces applications, nous pouvons citer la transmission de données, le traitement de la parole et, d'une manière générale, le filtrage des séries chronologiques.

BIBLIOGRAPHIE

[1] Lawrence R. Rabiner & Bernard Gold, *Theory and Application of Digital Signal Processing*. PRENTICE-HALL, INC. Englewood Cliffs, New Jersey, USA, 1975.

[2] M. Bellanger, *Traitement numérique du signal- Théorie et pratique*. Masson et CNET-ENST, Paris, France, 1981.

[3] B.Gold & K.L.Jordan, Jr, "A direct search procedure for desining finite duration impulse response filters," IEEE *Trans. Audio Electroacoust.* Vol. AU-17, pp. 33-36, USA, Mar. 1969.

[4] L.R.Rabiner, B.Gold & C.McConegal,"An approch to the approximation problem for nonrecursive digital filters," IEEE *Trans. Audio Electroacoust.* (Special issue on digital filtering) Vol. AU-18, pp.83-106, USA, June 1970.

[5] J.H.McClellan, T.W.parks, and L.R.Rabiner, "A computer program for designing optimum FIR linear phase digital filters," IEEE *Trans. Audio Electroacoust.*, Vol. AU-21, pp.506-526, USA, December 1973.

[6] James H. Mc Clellan & Thomas W. Parks, "A unified approch to the design of optimum FIR linear phase digital filters," IEEE *Trans. Circuit Theory*, Vol CT-20, No 6, pp.697-701, USA, March 1973.

[7] L.R.Rabiner, "Linear programming design of finite impulse response (FIR) digital filters," IEEE *Trans. Audio Electroacoust.*, Vol. AU-20, pp. 280-288, USA, Oct. 1972.

[8] M. Kunt, *Traitement numérique des signaux*. DUNOD, Paris, France, 1981.

[9] B. Picinbono, *Eléments de théorie du signal*. DUNOD, Paris, France, 1977.

[10] R.Boite et H.Leich, *Les filtres numériques*. Coll. Techn. et scienti. des télécom. Masson, Paris, France, 1980.

[11] Pierre Siohan & Acyl Benslimane, "Synthèse des filtres numériques non récursifs à phase linéaire et coefficients de longueur finie," *Annales de Télécom.*, 39, No 7-8, pp.307-322, France, 1984.

[12] Thomas W. parks & James H. Mc Clellan, "Chebyshev approximation for non recursive digital FIR filters with linear phase," IEEE *Trans. Circuit Theory*, Vol CT-19, No 2, pp.189-194, USA, March 1972.

[13] G.A.Watson, *Approximation theory and numerical methods.* John Wiley & Sons, Chichister, England, 1980.

[14] Hadscomb (Ed.), *Methods of numerical approximations.* (Lectures delivred at a summer school held at Oxford University). Ed. Hadscomb, USA, 1965.

[15] M. Hasler et J. Neirynck, *Les filtres éléctriques.* DUNOD, Paris, France, 1981.

[16] L.R.Rabiner, "Technics for designing finite duration impulse-response digital filters," IEEE *Trans. Communic. Tecnol.* Vol. COM-19, pp.188-195, USA, Apr. 1969.

[17] O.Hermann, "Design of nonrecursive digital filters with linear phase," *Electronic Lett.* pp. 228-229, USA, 1979.

[18] H.D.Helms, "Digital filters with equiripple or minimax responses," *IEEE Trans. Audio Electroacoust.*, Vol. AU-19, pp.87-93, USA, Mar. 1971.

[19] C. M. Lee & F. D. K. Roberts, "A comparison of algorithms for rational L_∞ approximation," *Mathematics of computation*, Vol. 27, No 121, pp. 111-121, USA, January 1973.

[20] Y.Chen & T.W.Parks, "Desig of FIR filters in the complex domain," IEEE *Trans. Acoust. Speech and Signal Process.*, Vol ASSP-35, No 2, pp. 144-153, USA, Feb. 1987.

[21] Ashraf Alkhairy, Kevin Christian & Jae Lim, "Design of FIR filters by complex Chebyshev approximation" Massachusetts institute of technology, Cambridge, IEEE 1991, pp. 1985-1988, USA, 1991.

[22] F.Dorosebeke, M.Hallen & Cl.Lefevre, *Programmation linéaire par l'exemple*. Editions Marketing, Paris, France, 1986.

[23] M. Sakarovitch, *Optimisation combinatoire -Graphes et programmation linéaire*. Editions Hermann, Paris, France, 1984.

[24] Narandra Paul Loomba & Efraim Turban, *Applied programming for management*. Holt, Rinehart & Winston, Inc. USA, 1974.

[25] Donald A. Pierre, *Optimization theory with applications*. Dover publications, Inc. New York, USA, 1986.

[26] Dusan M. Kodek, "Design of optimal finite word-length FIR digital filters using integer programming technics," IEEE *Trans. Acoust. Speech and Signal Process.*, Vol ASSP-28, pp. 304-308, USA, June 1980.

[27] N.B.Lawrance & A.C.Salazar, "Finite precision design of linear-phase FIR filters," *Bell Syst. Techn. J.*, No 9, pp. 1575-1598, USA, Nov.1980.

[28] F.Grenez, "Synthèse des filtres numériques non récursifs à coefficients quantifiés," *Annales de Télélcom.*, 34, No 1-2, pp.33-39, France, 1979.

[29] M. Sakarovitch, *Optimisation combinatoire -Optimisation discrète*. Editions Hermann, Paris, France, 1984.

[30] E.Avenhaus, "On the design of digital filters with coefficients of limited word length," IEEE *Tans. Acoust., Speech and Signal Process.*, Vol. ASSP-20, pp.206-212, USA, Aug. 1972.

[31] Dusan Kodek & Kenneth Steiglitz, "Comparison of optimal and local search methods for design finite word-length FIR digital filters" IEEE *Trans Circuits and Syst.*, Vol CAS-28, No 1, pp. 28-32, USA, Mar. 1981.

ANNEXE I : L'ALGORITHME DU SIMPLEXE

A-PRESENTATION THEORIQUE

A-1-Formulation d'un programme linéaire

Un *programme linéaire* est un problème du type [22,23]:

Maximiser z, tel que $z=\sum_{j=1}^{n} c_j x_j$ $\{x_j, j=1,\ldots,n\}$	Minimiser z, tel que $z=\sum_{j=1}^{n} c_j x_j$ $\{x_j, j=1,\ldots,n\}$
Sous les contraintes:	Sous les contraintes:
$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad , \quad i = 1,\ldots,m$	$\sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad , \quad i = 1,\ldots,m$ (A-1-a)
$x_j \geq 0 \quad , \quad j = 1,\ldots,n$	$x_j \geq 0 \quad , \quad j = 1,\ldots,n$ (A-1-b)
pour un problème à maximum	pour un problème à minimum

où les c_j, les a_{ij} et les b_j sont des constantes réelles. Les x_j (dites *variables structurelles*) sont des variables réelles.

Il est toujours possible de présenter les contraintes du problème (A-1) sous forme d'égalités en introduisant des *variables d'écart* t_i, on aura alors [22,23]

$\sum_{j=1}^{n} a_{ij} x_j + t_i = b_i \quad , \quad i=1,\ldots,m$	$\sum_{j=1}^{n} a_{ij} x_j - t_i = b_i \quad , \quad i=1,\ldots,m$
$x_j \geq 0 \quad , \quad j = 1,\ldots,n$	$x_j \geq 0 \quad , \quad j = 1,\ldots,n$ (A-2)
$t_i \geq 0 \quad , \quad i = 1,\ldots,m$	$t_i \geq 0 \quad , \quad i = 1,\ldots,m$
pour un problème à maximum	pour un problème à minimum

Si on tient compte du fait que ces variables d'écart peuvent être traitées comme les variables structurelles et en posant:

$$n1=n+m$$

un programme linéaire peut se formuler comme suit:

$$
\begin{array}{l}
\text{Maximiser (ou minimiser) } z, \text{ tel que } z = \sum_{j=1}^{n} c_j . x_j \\
\{x_j, j=1,\ldots,n1\} \\
\text{Sous les contraintes:} \\
\sum_{j=1}^{n1} a_{ij} x_j = b_i \ , \quad i = 1,\ldots,m \\
x_j \geq 0 \ , \quad j = 1,\ldots,n1
\end{array}
\qquad \text{(A-3)}
$$

On dit alors que le programme linéaire est mis sous sa forme standard.

Une autre façon de présenter un programme linéaire est la suivante. Posons:

$$
A=\begin{bmatrix}
a_{11}a_{12}\ldots\ldots a_{1n} & \overbrace{1\ 0\ \ldots\ldots 0}^{m} \\
a_{21}a_{22}\ldots\ldots a_{2n} & 0\ 1\ \ldots\ldots 0 \\
\vdots\ \ \vdots\ \ a_{ij}\ \ \vdots\ \ \vdots & \vdots \\
a_{m1}a_{m2}\ldots\ldots a_{mn} & 0\ 0\ \ldots\ldots 1
\end{bmatrix}, \quad
x=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n1} \end{bmatrix}, \quad
c=\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n1} \end{bmatrix}, \quad
b=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}
$$

le problème (A-3) peut être réécrit sous la forme:

$$
\max_x \text{ (ou } \min_x \text{)} \ z = c^t x \quad \text{t.q.} \quad Ax = b \qquad \text{(A-4)}
$$

A-2-Autres définitions [22]

-Un vecteur x est dit *solution* s'il satisfait à (A-1-a).

-Une solution est dite *solution réalisable* si elle satisfait à (A-1-b).

-Une *base* est tout ensemble de m variables telle que le determinant de la matrice a_{ij} associée à ces m variables soit différent de 0.

-Une *solution de base* est toute solution comprenant n-m variables nulles et telle que les m autres variables forment une base.

-Une solution de base est dite *solution de base réalisable* si elle satisfait à (A-1-b).

-Une solution est dite *solution optimale* si elle est solution de base réalisable et optimise z.

A-3-Résolution d'un problème de programmation linéaire

L'ensemble des solutions réalisables d'un problème de programmation linéaire détermine dans l'espace des variables structurelles un ensemble polyédrique convexe appelé domaine réalisable. La solution optimale occupe un sommet de ce polyèdre.

Les méthodes les plus utilisées pour résoudre un problème de programmation linéaire sont les suivantes [22,23,24,25]:

-Méthode graphique: basée sur l'étude graphique de z en fonction des variables structurelles x_j, elle ne peut être appliquée que lorsqu'il y a une ou deux variables seulement.

-Méthode de dénombrement des solutions de base: elle consiste à énumérer toutes les solutions de base réalisables ensuite d'en extraire celle qui donne z optimum. C'est une méthode très lente [22].

-Méthode du simplexe: cette méthode, très utilisée, permet de voir s'il y a au moins une solution réalisable et, dans l'affirmative, elle fournit une solution optimale après un certain nombre fini d'itérations [22,23,24].

B-L'ALGORITHME DU SIMPLEXE

La résolution d'un problème de programmation linéaire par la méthode du simplexe est présentée sous forme de tableaux successifs appelés *tableaux simplexe* [22]. Le tableau initial reprend les données du problème (contraintes mises sous forme d'égalités et fonction économique) en les disposant de la façon décrite par le tableau 1.

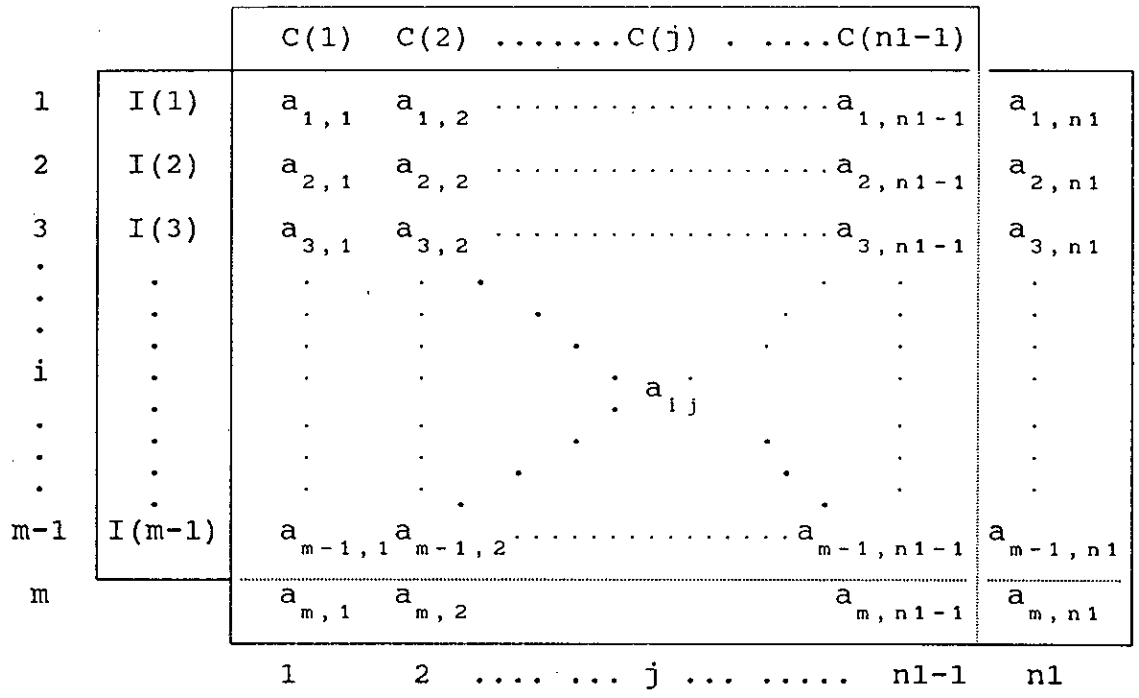
B-1-Le tableau simplexe

La première colonne, {I(i), i=1,...m-1}, contient les indices des variables formant une base. La dernière colonne $\{a_{i,n1}$, i=1,...,m-1} contient les valeurs prises par ces variables. Ces deux colonnes, sont initialisées, dans le premier tableau simplexe, à la solution de base réalisable de départ (voir § B-3). Il est à noter que les variables dans le tableau simplexe sont désignées par des indices.

La matrice $\{a_{ij}$, i=1,...,m-1 et j=1,...,n1-1} renferme les

coefficients des variables (structurelles et d'écart) dans les contraintes.

Tableau 1. Le tableau simplexe



La première ligne {C(j), j=1,...,n1-1} contient les coefficients des variables dans la fonction économique z.

La dernière ligne ($a_{m,j}$, j=1,...,n1-1) représente l'état du tableau (ligne test). Dans le tableau initial elle est initialisée aux valeurs suivantes:

$$a_{m,j}=-C(j)$$

Enfin, la quantité $a_{m,n1}$ contient la valeur de la fonction économique z à optimiser.

B-2-Organigramme de l'algorithme du simplexe (Fig. 1)

La théorie du simplexe [22,23,24,25] montre que, pour un problème à maximum, si la condition

$$a_{m,j} \leq 0 \qquad \forall\ j=1,...n1-1 \qquad \text{(B-1)}$$

($a_{m,j} \geq 0$ pour un problème à minimum)

est vérifié dans un tableau simplexe, alors la solution optimale est atteinte (on dit aussi que ce tableau est optimal).

Si la condition (B-1) n'est pas vérifiée dans le tableau, la

solution de base n'est pas optimale et on doit former une nouvelle solution de base: une variable sort de la base et est remplacée par une autre selon une règle bien determinée (voir organigramme de la figure 1). Un nouveau tableau est construit et est obtenu à partir du tableau précédent par des transformations qui sont décrites en detail dans l'organigramme. La procédure n'est arrêtée qu'après obtention d'une solution de base optimale, c'est-à-dire, qui vérifie la condition (B-1).

La solution optimale est donnée alors par [22,23,24]

$$\begin{cases} x(I(i))=a_{i,n1}, & i=1,\ldots,m-1 \\ x(j)=0 \quad \text{ailleurs} \end{cases} \qquad \text{(B-2)}$$

B-3-Initialisation du tableau simplexe

L'application du simplexe nécessite la connaissance d'une solution de base réalisable de départ. Or, une telle solution n'est pas toujours évidente. On peut avoir recours, dans ce cas, à deux méthodes qui permettent d'obtenir d'une façon automatique une solution de base réalisable de départ, il s'agit d'une méthode dite *méthode M* et de la méthode dite *méthode en deux phase* [22,23] (Fig. 2). Chacune de ces méthodes nécessite l'introduction de nouvelles variables dans le problème initial dites *variables artificielles* [22,23].

B-4-L'algorithme dual-simplexe

L'introduction de variables artificielles par la méthode décrite précédemment augmente considérablement le volume de calcul et ralentit, par conséquent, la convergence. Nous pouvons appliquer, sous une certaine condition, l'*algorithme dual-simplexe* [22,23] qui ne nécessite pas la connaissance préalable d'une solution de base réalisable de départ.

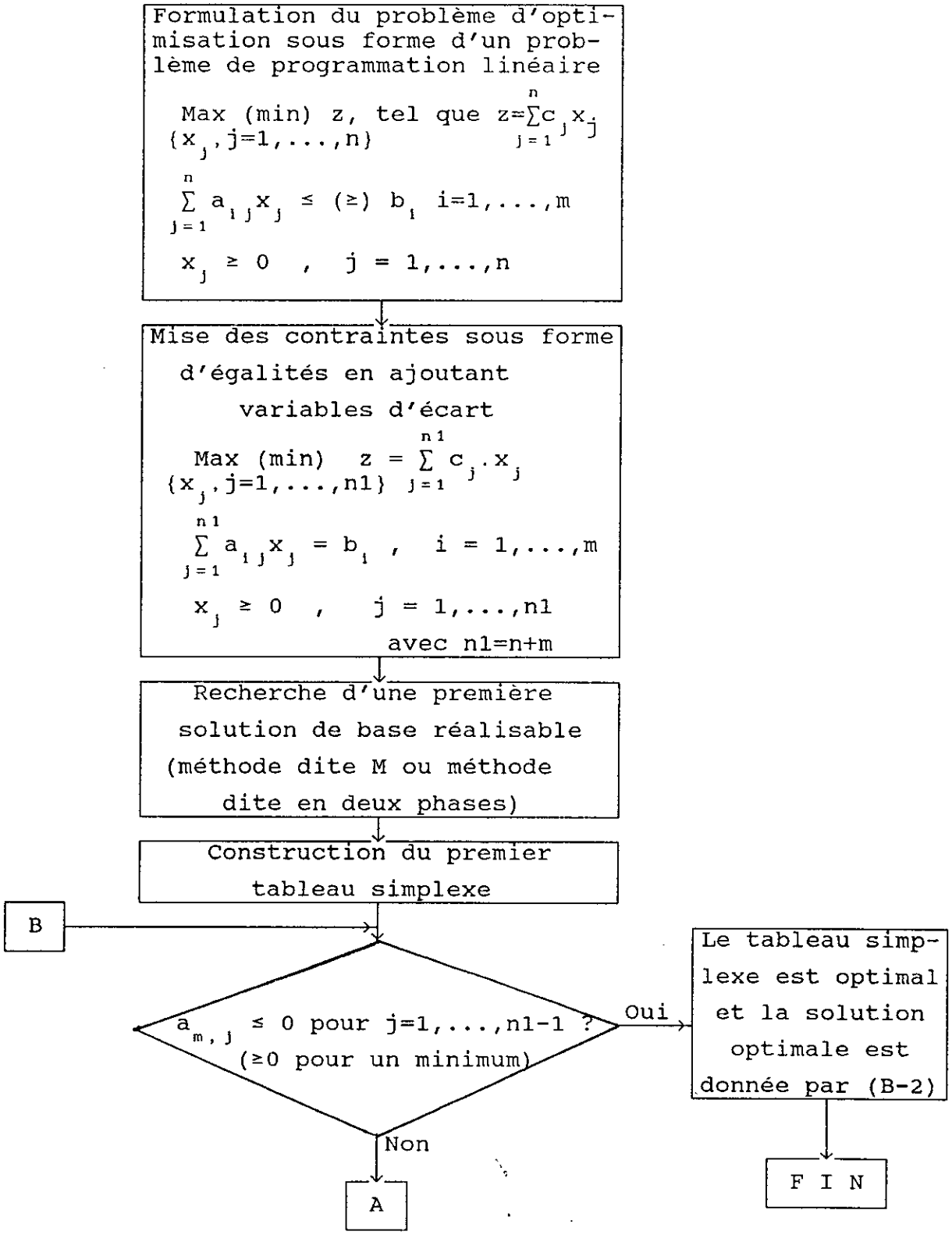
Par définition, le dual du problème de programmation linéaire formulé dans (A-3) est donné par [22,23]

$$
\begin{aligned}
&\text{Minimiser (ou maximiser) } z, \text{ tel que } z = \sum_{i=1}^{m} b_i . y_i \\
&(y_i, i=1,\ldots,m) \\
&\text{Sous les contraintes:} \\
&\sum_{i=1}^{n+m} a_{ij} y_i = b_j \quad , \quad j = 1,\ldots,n \\
&y_i \geq 0 \quad , \quad i = 1,\ldots,m
\end{aligned}
\qquad \text{(B-3)}
$$

Le dual étant lui-même un problème de programmation linéaire, on peut évidemment le traiter par l'algorithme du simplexe. L'algorithme résultant est appelé *dual-simplexe*.

L'algorithme dual-simplexe (Fig. 3) s'applique à tout programme linéaire possédant une solution de base non réalisable telle que:

$$a_{m,j} \leq 0 \quad \text{pour } j=1,\ldots n1-1 \qquad \text{(B-4)}$$


Formulation du problème d'optimisation sous forme d'un problème de programmation linéaire
Max (min) z, tel que $z=\sum_{j=1}^{n} c_j x_j$
$\{x_j, j=1,\ldots,n\}$
$\sum_{j=1}^{n} a_{ij} x_j \leq (\geq) b_i \quad i=1,\ldots,m$
$x_j \geq 0 \quad , \quad j = 1,\ldots,n$
Mise des contraintes sous forme d'égalités en ajoutant variables d'écart
Max (min) $z = \sum_{j=1}^{n1} c_j . x_j$
$\{x_j, j=1,\ldots,n1\}$
$\sum_{j=1}^{n1} a_{ij} x_j = b_i \quad , \quad i = 1,\ldots,m$
$x_j \geq 0 \quad , \quad j = 1,\ldots,n1$
avec n1=n+m
Recherche d'une première solution de base réalisable (méthode dite M ou méthode dite en deux phases)
Construction du premier tableau simplexe
B
$a_{m,j} \leq 0$ pour j=1,...,n1-1 ?
(≥0 pour un minimum)
Oui
Le tableau simplexe est optimal et la solution optimale est donnée par (B-2)
Non
A
F I N

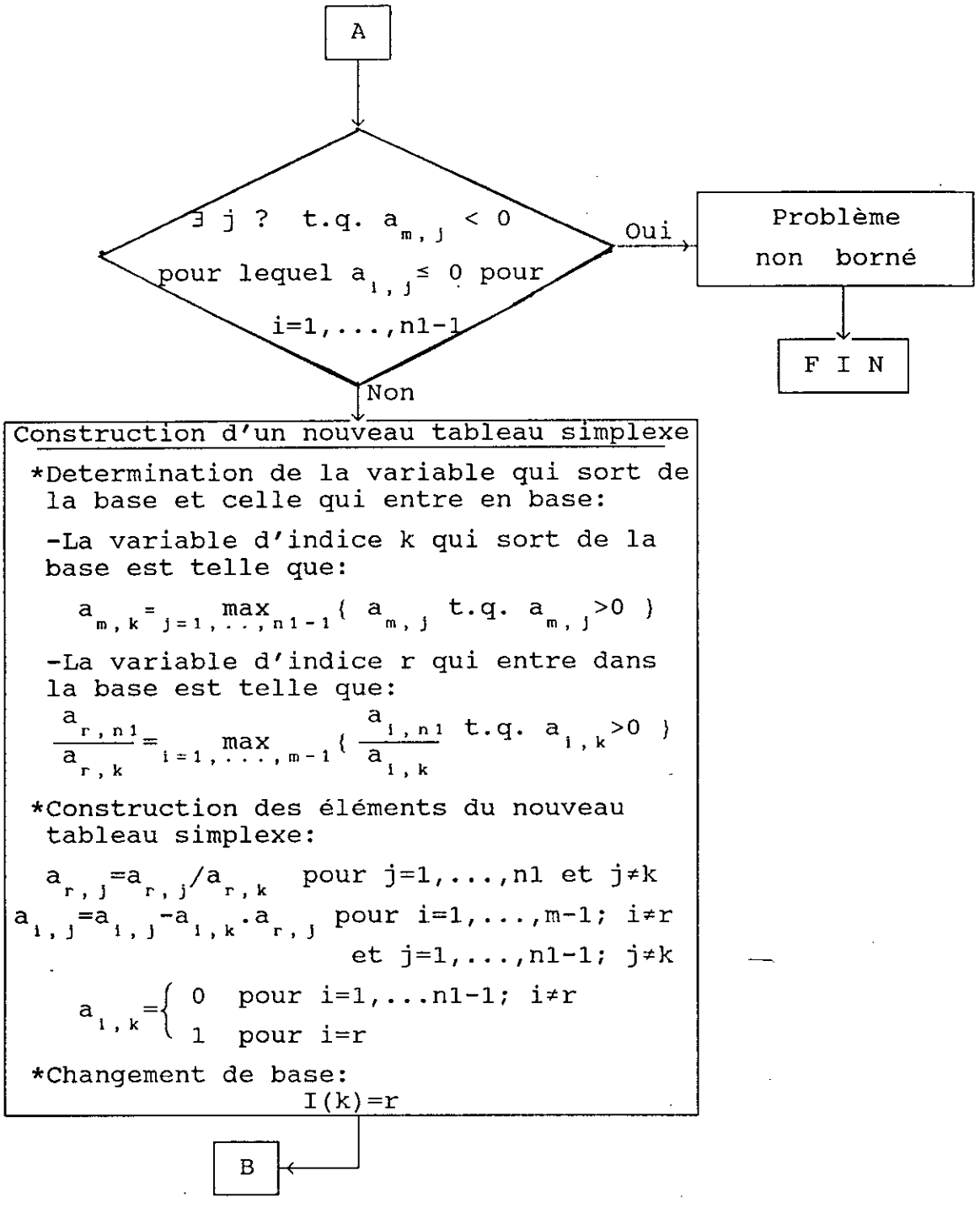


Fig. 1.Organigramme de l'algorithme du simplexe

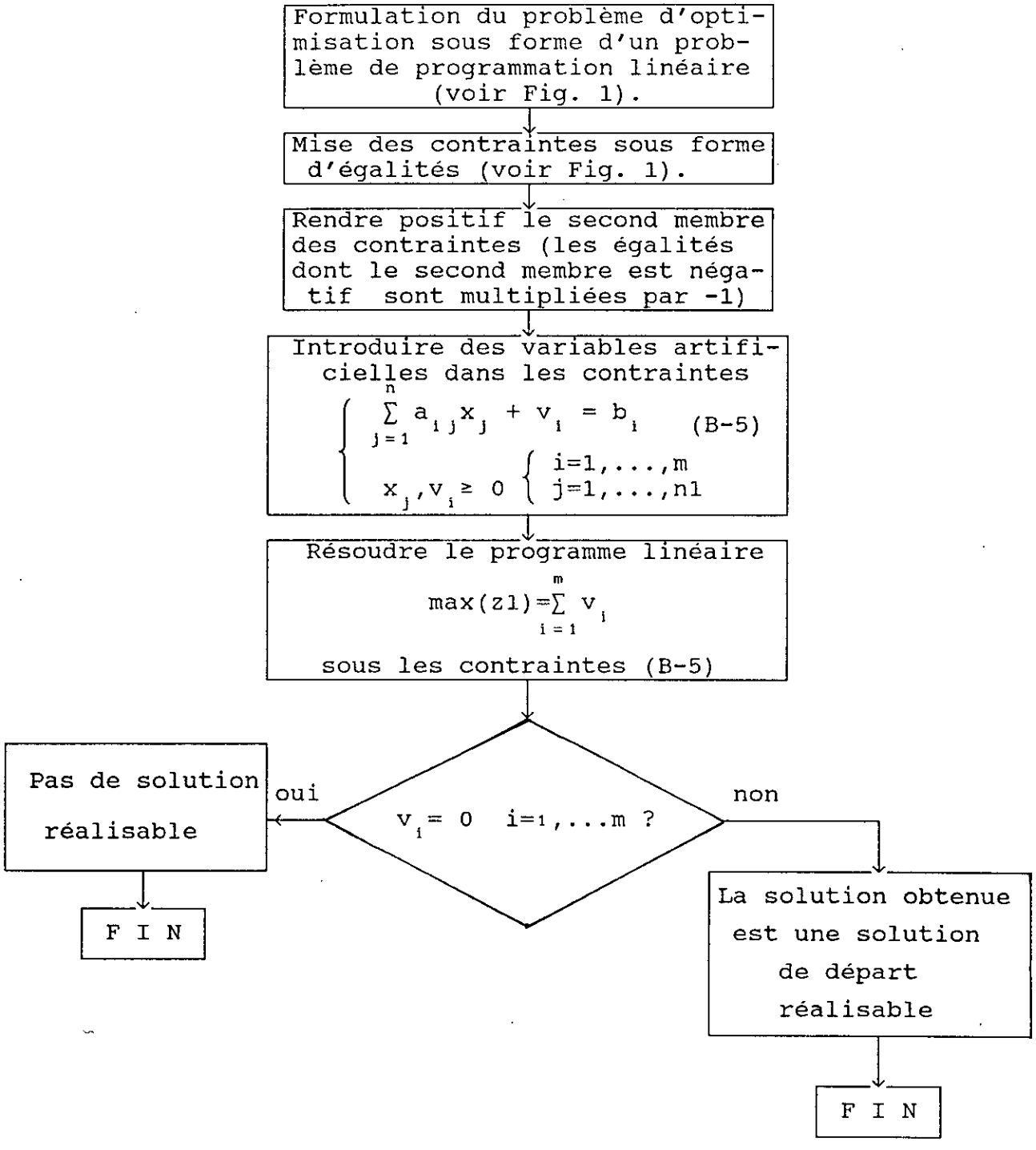
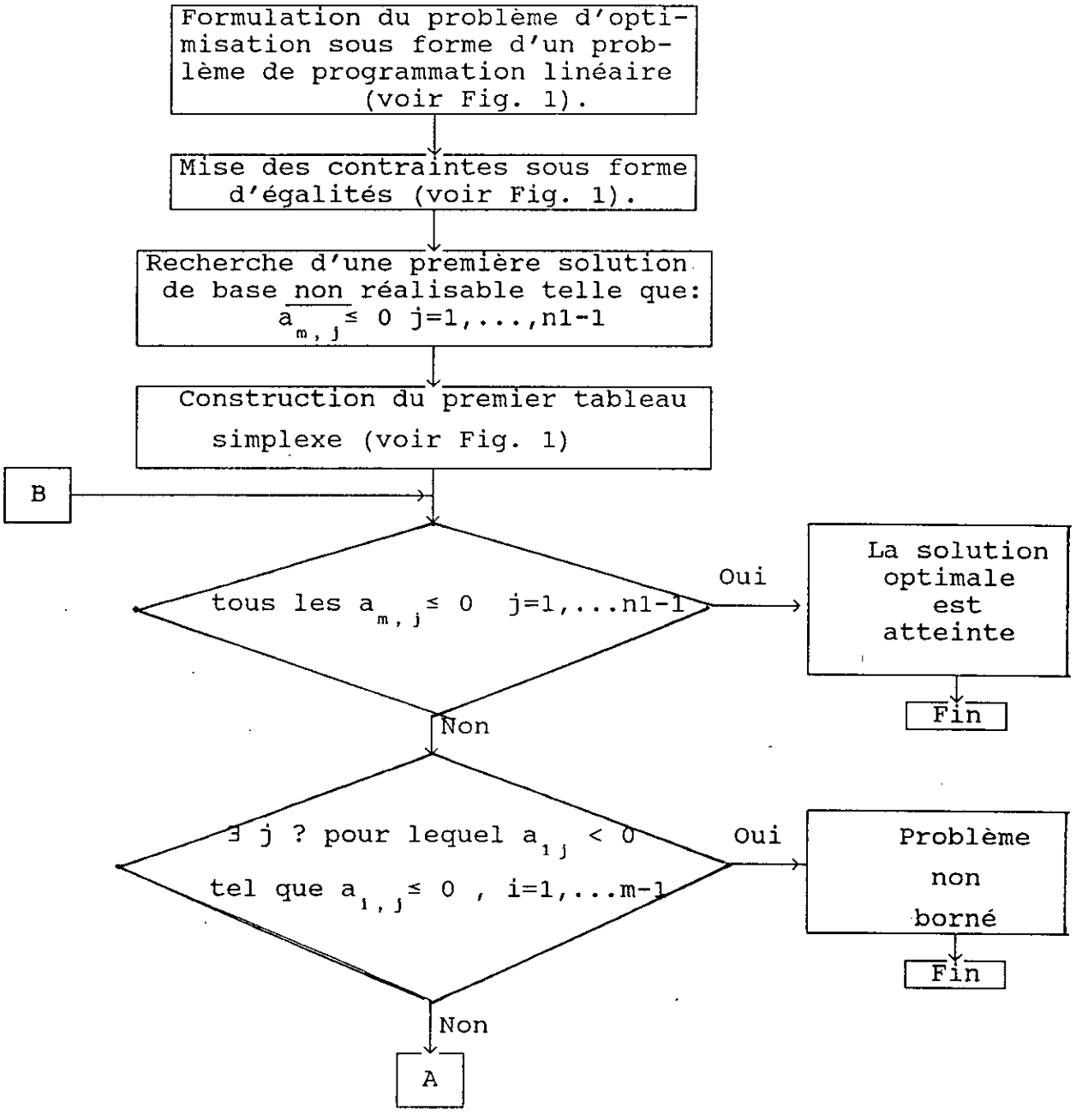


Fig. 2-Organigramme de la méthode dite en deux phases pour l'obtention d'une solution de base de départ.


Formulation du problème d'optimisation sous forme d'un problème de programmation linéaire (voir Fig. 1).
Mise des contraintes sous forme d'égalités (voir Fig. 1).
Recherche d'une première solution de base non réalisable telle que: $a_{m,j} \leq 0$ j=1,...,n1-1
Construction du premier tableau simplexe (voir Fig. 1)
B
tous les $a_{m,j} \leq 0$ j=1,...n1-1
Oui
La solution optimale est atteinte
Fin
Non
∃ j ? pour lequel $a_{1j} < 0$ tel que $a_{i,j} \leq 0$, i=1,...m-1
Oui
Problème non borné
Fin
Non
A

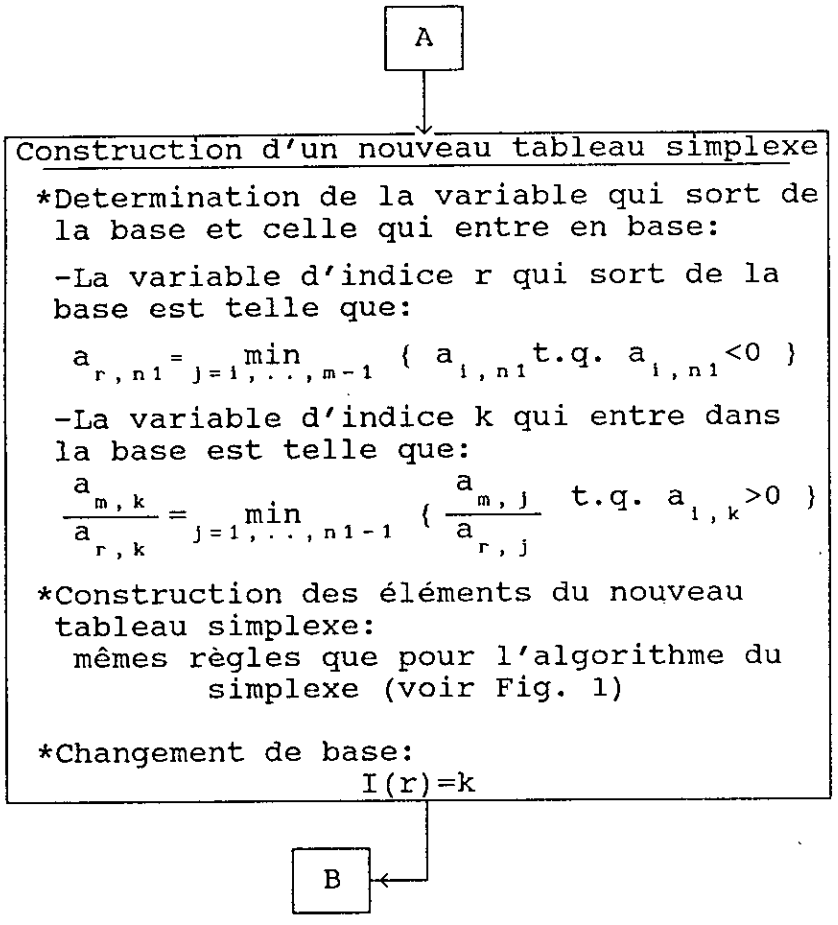


Fig. 3. Organigramme de l'algorithme dual-simplexe.

ANNEXE II : PROGRAMMES

A-PROGRAMME FASTSIMPLEX

```
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C     *                      PROGRAMME FASTSIMPLEX                        *
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C     CE PROGRAMME UTILISE L'ALGORITHME DSUAL-SIMPLEXE, IL CALCULE EN
C     UN TEMPS MINIMUM LES COEFFICIENTS OPTIMAUX D'UN FILTRE NUMERIQUE
C     RIF A PHASE LINEAIRE AU SENS DE LA NORME DE CHEBYSHEV.
C
C     LES DONNEES D'ENTREE SONT LUES A PARTIR D'UN FICHIER DE DONNEES
C     SPECIFIE PAR SON NOM, LES RESULTATS SONT ALORS SAUVEGARDES, EN
C     CAS DE NECESSITE, DANS UN FICHIER DONT ON DONNERA LE NOM.
C
C     OPTION SUPPLEMENTAIRE: OPTIMISATION DE LA REPONSE INDICIELLE DU
C     FILTRE NUMERIQUE.
C
      IMPLICIT REAL*4 (A-H,O-Z)
      REAL*4 FI(8),FS(8),DELTA(8),ERR(8)
      REAL*4 C(512),A(512)
      REAL*4 X(150,300)
      REAL*4 XO(256),CB(256),H(64),B(64),A1(128)
      REAL*4 D(8),W(8),DD(8)
      REAL*4 DN(500),WN(500),QN(500)
      REAL*4 HF(500),LHF(500)
      REAL*4 CS(64,500)
      INTEGER NFI(8),NFS(8)
      INTEGER NFEXT(128),NFEXT1(128),IB(256)
      CHARACTER*12 FICH

      PI=4.*ATAN(1.)

C     NOMBRE MAXIMUM DITERATIONS
      IEX1=10

C     DENSITE DE GRILLE
      LGRID=16

C     LECTURE DES DONNEES:
C     ********************

      WRITE(*,*)'Introduire le fichier gabarit,'
      READ(*,5)FICH
5     FORMAT(A)
      OPEN(UNIT=10,FILE=FICH,STATUS='OLD')

      WRITE(*,*)'DONNEES DU GABARIT'
      READ(10,*)NBAND
      WRITE(*,*)'NBAND=',NBAND
      DO 10 K=1,NBAND
      READ(10,*)FI(K),FS(K)
```

```
      WRITE(*,*)'FI(K)=,FS(K)=',FI(K),FS(K)
      READ(10,*)D(K)
      WRITE(*,*)'D(K)=',D(K)
      READ(10,*)DD(K)
      WRITE(*,*)'DD(K)=',DD(K)
10    CONTINUE

      DDMIN=DD(1)
      DO 15 K=2,NBAND
      IF(DD(K).GT.DDMIN) GOTO 15
      DDMIN=DD(K)
      MIN=K
15    CONTINUE
      DO 20 K=1,NBAND
      W(K)=DDMIN/DD(K)
20    CONTINUE

21    CONTINUE

      WRITE(*,*)' '
      WRITE(*,*)'Introduire l ordre du filtre '
      READ(*,*)NFILT
      NF=(NFILT-1)/2

C     DISCRETISATION DU PROBLEME
C     **************************

      LG=LGRID*NF

      DO 25 K=1,NBAND
      NFI(K)=INT(FI(K)*LG/.5+.5)
      NFS(K)=INT(FS(K)*LG/.5+.5)
25    CONTINUE
      DO 33 K=1,NBAND
      DO 30 J=NFI(K),NFS(K)
      DN(J)=D(K)
      WN(J)=W(K)
30    CONTINUE
33    CONTINUE

C     OPTION SUPPLEMENTAIRE: OPTIMISATION DE LA REPONSE INDICIELLE
C     ************************************************************

C     WRITE(*,*)' '
      WRITE(*,*)'Optimisation de la réponse indicielle:'
      WRITE(*,*)'1-Oui'
      WRITE(*,*)'2-Non'
      READ(*,*)ind
      MOSC=0
      IF(ind.eq.2) GOTO 24
      MOSC=NF-1
      WRITE(*,*)' '
      WRITE(*,*)'Introduire la limite des oscillations'
      READ(*,*)DOSC
24    CONTINUE
```

```
C       FORMULATION D'UN PROBLEME EQUIVALENT (SELON LA PARITE DE NFILT)
C       ***************************************************************

        NFILTRE=NFILT/2
        NFILTRE=2*NFILTRE
        IF(NFILTRE.NE.NFILT) GOTO 36
        LPAR=0
        GOTO 37
36      LPAR=1
37      CONTINUE

        IF(LPAR.EQ.1) GOTO 100
        NF=NFILT/2-1
        DO 70 K=0,LG-1
        DN(K)=DN(K)/COS(.5*PI*K/LG)
        WN(K)=WN(K)*COS(.5*PI*K/LG)
70      CONTINUE
        DN(LG)=0
100     CONTINUE

C       CALCUL DE LA FONCTION CS(NF,LG)
        DO 40 J=0,LG
        DO 35 I=0,NF
        CS(I,J)=COS(PI*I*J/LG)
35      CONTINUE
40      CONTINUE

C       CALCUL DE LA FONCTION QN(J)
        DO 50 J=0,LG
        QN(J)=DN(J)
        DO 50 I=0,NF
50      QN(J)=QN(J)+10.*CS(I,J)

C       INITIALISATION DES FREQUENCES EXTREMALES
C       ****************************************

        J=0
        DO 120 K=1,NBAND
        J=J+1
        NFEXT1(J)=NFI(K)
        DO 110 I=NFI(K)+1,NFS(K)-1,LGRID
        J=J+1
        NFEXT1(J)=I
110     CONTINUE
        J=J+1
        NFEXT1(J)=NFS(K)
120     CONTINUE
        NEX1=J

C       CONSTRUCTION DU 1ER TABLEAU SIMPLEX
C       ***********************************

       IEX=0
        WRITE(*,122)
122     FORMAT(//////////////,20X,'E X E C U T I O N   E N   C O U R S'
     * ////////////)
125     CONTINUE
```

```
          IEX=IEX+1
          NEX=NEX1
          DO 130 I=1,NEX
          NFEXT(I)=NFEXT1(I)
130       CONTINUE

          CALL SBR1(NF,CS,WN,QN,NEX,NFEXT,
      *   ITAB,JTAB,X,XO,MOSC,DOSC)

C         LE NOYAU DUAL-SIMPLEXE
C         **********************

          DO 410 J=1,JTAB
          C(J)=0.
410       CONTINUE
          C(NF+2)=1.
          DO 420 I=1,ITAB
          IB(I)=I+NF+2
          CB(I)=C(IB(I))
420       CONTINUE
          do 430 i=1,itab
430       x(i,jtab+1)=xo(i)
          do 440 j=1,jtab
440       x(itab+1,j)=-c(j)
          x(itab+1,jtab+1)=zo
          itab=itab+1
          jtab=jtab+1

         CALL SBR2(X,C,CB,IB,ITAB,JTAB,JOIE,ZO)

         IF(JOIE.EQ.0) GOTO 990

C         EVALUATION DE L'ERREUR ET DETERMINATION DES NOUVELLES
C         *****************************************************
C         FREQUENCES EXTREMALES
C         *********************

C         CALCUL DES COEFFICIENTS A(J) (X(J) dans le texte)
          DO 450 J=1,JTAB-1
450       A(J)=0
          DO 460 I=1,ITAB-1
460       A(IB(I))=X(I,jtab)
          E=A(NF+2)

C         CALCUL DE LA REPONSE FREQUENCIELLE
          DO 470 I=1,NF+1
470       A(I)=A(I)-10.
          DO 480 J=0,LG
          HF(J)=0.
          DO 475 I=1,NF+1
475       HF(J)=HF(J)+A(I)*CS(I-1,J)
480       CONTINUE

C         DETERMINATION DES NOUVELLES FREQUENCES EXTREMALES

          CALL SBR3(NBAND,NFI,NFS,HF,NEX1,NFEXT1,ERR,DN,WN)
```

```
C       TEST DE L'OPTINMALITE
C       **********************

        IF(NEX1.NE.NEX) GOTO 125
        IF(IEX.GE.IEX1) GOTO 500
        DO 490 I=1,NEX1
        IF(NFEXT1(I).NE.NFEXT(I)) GO TO 125
490     CONTINUE
500     CONTINUE

C       SECTION DE SORTIE
C       *****************

C       CALCUL DES COEFFICIENTS

        IF(LPAR.EQ.0) GOTO 580
        DO 520 I=0,NF-1
520     H(I)=A(NF-I+1)/2
        H(NF)=A(1)
 DO 550 I=NF+1,NFILT-1
550     H(I)=H(NFILT-1-I)
        GOTO 590

580     CONTINUE
        B(1)=A(1)+.5*A(2)
        DO 585 I=2,NF
585     B(I)=.5*(A(I)+A(I+1))
        B(NF+1)=.5*A(NF+1)
        NF=NF+1
        DO 586 I=0,NF-1
586     H(I)=.5*B(NF-I)
        DO 587 I=NF,NFILT-1
587     H(I)=H(NFILT-1-I)

590     CONTINUE

        WRITE(*,*)'RESULTATS:'

        IF(LPAR.EQ.0) GOTO 610
        DO 600 I=0,NF-1
        WRITE(*,602)I,H(I),NFILT-1-I
602     FORMAT(5X,' H(',I3,') = ',E15.8,' = H(',I3,')')
600     CONTINUE
        WRITE(*,603)NF,H(NF)
603     FORMAT(10X,'H(',I3,') = ',E15.8)
        GOTO 607

610     CONTINUE
        DO 605 I=0,NF-1
        WRITE(*,604)I,H(I),NFILT-1-I
604     FORMAT(5X,' H(',I3,') = ',E15.8,' = H(',I3,')')
605     CONTINUE

607     CONTINUE
        WRITE(*,*)'Erreur optimale E=',E
        WRITE(*,*)'Temps d execution NTEMPO=',NTEMPO
        WRITE(*,*)'ERR=',(ERR(I),I=1,NBAND)
```

```
C       COMPLEMENT: ETUDE DU REGIME TRANSITOIRE
C       ***************************************

C       CALCUL DE LA REPNSE INDICIELLE
        DO 1001 m=0,NFILT-1
        a1(m)=0.
        DO 501 i=0,m
501     a1(m)=a1(m)+h(i)
1001    CONTINUE

C       DETERMINATION DU MAXIMUM DES OSCILLATIONS DE LA REPONSE
C       INDICIELLE
        A1MAX=0.
        do 2001 m=0,NF-1
        IF(ABS(A1(M)).LT.A1MAX) GOTO 2001
        A1MAX=ABS(A1(M))
2001    CONTINUE
        DO 2002 M=NF+2,NFILT-1
        IF(ABS(A1(M)-A1(NFILT-1)).LT.A1MAX) GOTO 2002
        A1MAX=ABS(A1(M))
2002    CONTINUE
        WRITE(*,*)'ONDULATION MAX.:',A1MAX

       WRITE(*,591)
591    FORMAT(//' 1-SAUVER '/' 2-ABANDONNER'
     * //' CHOISIR UN NUMERO')
        READ(*,*)IQ
        IF(IQ.EQ.2) GOTO 665
        WRITE(*,*)'DONNER LES NOMS DES FICHIERS:'
        WRITE(*,*)'-REPONSE IMPULSIONNELLE:'
        READ(*,592)FICH
592     FORMAT(A200)
        OPEN(1,FILE=FICH,STATUS='NEW')

        WRITE(1,*)nfilt
        DO 650 I=0,NFilt-1
        WRITE(1,*)H(I)
650     CONTINUE
        WRITE(1,*)E
        WRITE(1,*)NTEMPO

665     CONTINUE
        WRITE(*,670)
670     FORMAT(/////' 1-CHANGER NFILT'/' 2-FIN'/////
     * ' CHOISIR UN NUMERO'////)
        READ(*,*)IP
        IF(IP.EQ.1) GOTO 21
        GOTO 999
990     CONTINUE
        PRINT*,'                    *** PROBLEME NON BORNE ***'
        GOTO 665
999     CONTINUE
        STOP
        END
```

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C       SBR1:CHARGEMENT DU PREMIER TABLEAU SIMPLEX
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        SUBROUTINE SBR1(NF,CS,WN,QN,NEX,NFEXT,
     *  ITAB,JTAB,X,XO,MOSC,DOSC)
C
C       CE SOUS-PROGRAMME EST DESTINE A CALCULER LES COEFFICIENTS
C       DES VARIABLES DANS LES COTRAINTES
C
        IMPLICIT REAL*4(A-H,O-Z)
        INTEGER NFEXT(128)
        REAL*4 X(150,300),WN(500),QN(500)
        REAL*4 CS(64,500)
        REAL*4 XO(150)

        DO 1001 I=1,itab
        xo(i)=0
        DO 1000 j=1,jtab
        x(i,j)=0
1000    continue
1001    continue

C       1<=I<=NEX  1<=J<=NF+1

        DO 1100 I=1,NEX
        DO 1010 J=1,NF+1
        X(I,J)=CS(J-1,NFEXT(I))
1010    CONTINUE
        X(I,NF+2)=-1./WN(NFEXT(I))
        XO(I)=QN(NFEXT(I))
1100    CONTINUE

          DO 2132 M=0,MOSC-1
          X(NEX+M+1,NF+2)=0.
          XO(NEX+M+1)=2.*dosc+10.*(M+1)
          DO 2120 J=NF+1-M,NF+1
          X(NEX+M+1,J)=1.
2120      continue
2132      continue

          NNEX=NEX+MOSC

C         NEX<=I<=2*NEX

        DO 1300 I=NNEX+1,NNEX+NEX
        DO 1200 J=1,NF+1
        X(I,J)=-X(I-NNEX,J)
1200    CONTINUE
        X(I,NF+2)=X(I-NNEX,NF+2)
        XO(I)=-XO(I-NNEX)
1300    CONTINUE

        DO 21150 I=NNEX+NEX+1,2*NNEX
        DO 21100 J=1,NF+1
        X(I,J)=-X(I-NNEX,J)
21100    CONTINUE
```

```
          X(I,NF+2)=X(I-NNEX,NF+2)
          XO(I)=4.*dosc-XO(I-NNEX)
21150      CONTINUE

C         NEX+1<=I<=2*NEX  1<=J<=NF+1

            ITAB=2*NNEX

            JTAB=ITAB+NF+2

C       CHARGEMENT DE LA MATRICE X(I,J) POUR 1<I<ITAB & NF+3<J<NF+2+ITAE

          DO 2500 I=1,ITAB
          DO 2500 J=NF+3,JTAB
          IF(I.EQ.(J-NF-2)) GOTO 2400
          X(I,J)=0.0
          GOTO 2500
2400      X(I,J)=1.
2500      CONTINUE

          RETURN
          END

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C                 SBR2 : ALGORITHME DU SIMPLEX
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

          SUBROUTINE SBR2(X,C,CB,IB,ITAB,JTAB,JOIE,ZO)
C
C         CE SOUS-PROGRAMME EST UNE TRADUCTION DE L'ALGORITHME
C         DUAL-SIMPLEXE

          IMPLICIT REAL*4(A-H,O-Z)
          REAL*4 X(150,300)
          REAL*4 C(512),D(512)
          REAL*4 AA
          REAL*4 CB(256),XO(256)
          INTEGER IB(256)

          ITER=0

C         TEST DE L OPTIMALITE

4750       ITER =ITER

           DO 4800 I=1,ITAB-1
           IF(X(I,JTAB).GE.0) GOTO 4800
           GOTO 4850
4800       CONTINUE
           GOTO 5700

4850       CONTINUE
           DO 4950 I=1,ITAB-1
           IF(X(I,JTAB).GE.0) GOTO 4950
           DO 4900 J=1,JTAB-1
           IF(X(I,J).LT.0) GOTO 4950
4900       CONTINUE
```

```
         GOTO 5600
4950     CONTINUE

C        CHANGEMENT DE BASE

         DMIN=2.E+22
         DO 5300 I=1,ITAB-1
         IF((X(I,JTAB).GE.0.).OR.(X(I,JTAB).GT.DMIN)) GOTO 5300
         DMIN=X(I,JTAB)
         NR=I
5300     CONTINUE
         DMIN=2.E+22
         DO 5400 J=1,JTAB-1
         IF((X(NR,J).GE.0).OR.((X(ITAB,J)/X(NR,J)).GT.DMIN)) GOTO 5400
         DMIN=X(ITAB,J)/X(NR,J)
         NK=J
5400     CONTINUE

C        CONSTRUCTION DU NOUVEAU TABLEAU SIMPLEX

c         WRITE(*,*)'X(NR,NK)=',X(NR,NK)
         do 6560 J=1,Jtab
         IF(J.EQ.NK) GOTO 6560
         X(NR,J)=X(NR,J)/X(NR,NK)
         DO 6550 I=1,ITAB
         if(i.eq.nr) goto 6550
         AA=X(I,NK)*X(NR,J)
         X(I,J)=X(I,J)-AA
6550     continue
6560     continue
         DO 6570 I=1,ITAB
6570     X(I,NK)=0.
         X(NR,NK)=1.
         cb(nr)=C(nk)
         ib(nr)=nk
         goto 4750

5600     JOIE=0
         GOTO 5800
5700     JOIE=1
         zo=x(itab,jtab)
5800     CONTINUE
         RETURN
         END

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C        SBR3:CALCUL ET ANALYSE DE LA REPONSE FREQUENCIELLE
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

         SUBROUTINE SBR3(NBAND,NFI,NFS,HF,NEX1,NFEXT1,ERR,DN,WN)
C
C        CE PROGRAMME EST DESTINE A EVALUER LE NOMBRE ET LES VALEURS
C        EXTREMALES
C
         IMPLICIT REAL*4(A-H,O-Z)
         REAL*4 HF(500),DHF(500)
         REAL*4 ER(500),DN(500),WN(500),ERR(8)
```

```
      INTEGER NFI(8),NFS(8),NFEXT1(128)

      DO 6402 I=1,NBAND
      ERR(I)=0.
      DO 6401 J=NFI(I),NFS(I)
      ER(J)=ABS(WN(J)*(HF(J)-DN(J)))
      IF(ER(J).LT.ERR(I)) GOTO 6401
      ERR(I)=ER(J)
6401  CONTINUE
6402  CONTINUE

C     DETERMINATION DES EXTREMUMS
      DO 6430 K=1,NBAND
      DO 6450 J=NFI(K)+1,NFS(K)-1
      DHF(J)=HF(J+1)-HF(J-1)
6450  CONTINUE
6430  CONTINUE
      J=0
      DO 6600 K=1,NBAND
      J=J+1
      NFEXT1(J)=NFI(K)
      DO 6500 I=NFI(K)+2,NFS(K)-2
      IF((DHF(I-1)*DHF(I+1)).GT.0) GOTO 6500
      J=J+1
      NFEXT1(J)=I
6500  CONTINUE
      J=J+1
      NFEXT1(J)=NFS(K)
6600  CONTINUE
      NEX1=J
      RETURN
      END
```

B-PROGRAMME LOCALSEARCH

```
C       *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C       *  QUANTIFICATION SUB-OPTIMALE PAR RECHERCHE LOCALE     *
C       *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C
C       Ce programme permet l'obtention d'une solution sub-optimale
C       en partant d'une solution arrondie. Trois types d'optimisa-
C       tions sont offertes:
C           -Optimisation1: un seul coefficient change à la fois.
C           -Optimisation2: deux coefficients changent à la fois.
C           -OPtimisation3: trois coefficients changent à la fois.
C       Pour chaque type d'optimisation, les coefficients qui chang-
C       ent peuvent prendre toutes les valeurs fixées par un voisi-
C       nage au choix.
C       Les données d'entrée sont le gabarit et les coefficients
C       avec l'erreur optimale en précision infinie.
C       Ensuite le nombre de bits est demandé.
```

```
C       A la sortie les coefficients sub-optimaux sont donnés avec
C       l'erreur maximale correspondante.

      IMPLICIT REAL*4 (A-H,O-Z)
      DIMENSION H(128),HA(128),NHA(128)
      DIMENSION HC(128),HO(128)
      DIMENSION D(8),DD(8),W(8)
      DIMENSION FI(8),FS(8)
      DIMENSION DN(500),WN(500)
      INTEGER NFI(8),NFS(8)
      CHARACTER*12 FICH

      PI=4.*ATAN(1.)

C     LECTURE DU GABARIT
C     ******************

      WRITE(*,*)'DONNER LE FICHIER DU GABARIT,'
      READ(*,5)FICH
5     FORMAT(A)
      OPEN(2,FILE=FICH,STATUS='OLD')

      READ(2,*)NBAND
      READ(2,*)(FI(J),FS(J),D(J),DD(J),J=1,NBAND)

        DDMIN=DD(1)
        DO 15 K=2,NBAND
        IF(DD(K).GT.DDMIN) GOTO 15
        DDMIN=DD(K)
        MIN=K
15      CONTINUE
        DO 17 K=1,NBAND
        W(K)=DDMIN/DD(K)
17      CONTINUE


C     LECTURE DE LA REPONSE TEMPORELLE
C     ********************************

1     CONTINUE
      WRITE(*,*)'DONNER LE FICHIER DES COEFFICIENTS'
      READ(*,10)FICH
10    FORMAT(A)
      OPEN(1,FILE=FICH,STATUS='OLD')
      READ(1,*)NFILT
      READ(1,*)(H(I),I=0,NFILT-1)
      READ(1,*)EI

C     DISCRETISATION DU GABARIT
C     *************************

      LGRID=16
      LG=LGRID*NFILT/2

      DO 1100 K=1,NBAND
      NFI(K)=INT(FI(K)*LG/.5)
      NFS(K)=INT(FS(K)*LG/.5)
```

```
      DO 1000 J=NFI(K),NFS(K)
      DN(J)=D(K)
      WN(J)=W(K)
1000  CONTINUE
1100  CONTINUE

C     CHOIX DU NOMBRE DE BITS
C     ***********************

20    CONTINUE
      WRITE(*,*)'INTRODUIRE LE NOMBRES DE BITS:'
      READ(*,*)NBIT

C     QUANTIFICATION PAR ARRONDI
C     **************************

c        -Calcul des coefficients arrondis

      BIT=2**(NBIT-1)

      DO 110 I=0,NFILT-1
      IF(H(I).LT.0.) GOTO 110
      NHA(I)=INT(H(I)*BIT)
      HA(I)=FLOAT(NHA(I)/BIT)
110   CONTINUE

      DO 120 I=0,NFILT-1
      IF(H(I).GE.0) GOTO 120
      NHA(I)=INT(H(I)*BIT+.5)
      HA(I)=FLOAT(NHA(I))/BIT
120   CONTINUE

c         -Calcul de l'erreur d'arrondi

      CALL ERR(NFILT,HC,NBAND,NFI,NFS,DN,WN,PI,EC)

C     QUANTIFICATION PAR RECHERCHE LOCALE
C     ***********************************

135   CONTINUE
      WRITE(*,140)
140   FORMAT(////' 1-OPTIMISATION LOCALE'/' 2-CHANGER NBIT'/' 3-FIN'
     * //' CHOISIR UN NUMERO,')
      READ(*,*)IA
      IF(IA-2)150,20,3000
150   CONTINUE
      WRITE(*,200)
200   FORMAT(////' 1-OPTIMISATION1'/' 2-OPTIMISATION2'
     * /' 3-OPTIMISATION3'//' CHOISIR UN NUMERO,')
      READ(*,*)IP
      IF(IP-2)300,400,500

C     OPTIMISATION1
C     =============

300   CONTINUE
```

```
      WRITE(*,*)'VOISINAGE='
      READ(*,*)NVOI

C     GENERATION D'UNE COMBINAISON:

      EO=EA
      DO 370 I=0,NFILT/2
      DO 320 L=0,NFILT-1
320   HO(L)=HA(L)
      DO 360 K=-NVOI,NVOI
      DO 350 J=0,NFILT-1
350   HC(J)=HA(J)
      HC(I)=HA(I)+K/BIT

C     CALCUL DE L'ERREUR

      CALL ERR(NFILT,HC,NBAND,NFI,NFS,DN,WN,PI,EC)

C     TEST

      IF(EC.GT.EO) GOTO 360
      EO=EC
      HO(I)=HC(I)
360   CONTINUE
370   CONTINUE
      GOTO 599

C     OPTIMISATION2
C     =============

400   CONTINUE

      WRITE(*,*)'VOISINAGE='
      READ(*,*)NVOI

C     GENERATION D'UNE COMBINAISON

      DO 410 I=0,NFILT/2
      HO(I)=HA(I)
410   CONTINUE
      EO=EA

      DO 420 I=0,NFILT/2
      HC(I)=HA(I)
420   CONTINUE

      DO 470 I=0,NFILT/2
        DO 460 II=-NVOI,NVOI
          DO 450 J=I+1,NFILT/2
            DO 440 JJ=-NVOI,NVOI
            HC(I)=HA(I)+II/BIT
            HC(J)=HA(J)+JJ/BIT

C     CALCUL DE L'ERREUR

      CALL ERR(NFILT,HC,NBAND,NFI,NFS,DN,WN,PI,EC)
```

```
C       TEST

        IF(EC.GT.EO) GOTO 440
        EO=EC
        HO(I)=HC(I)
        HO(J)=HC(J)

440           CONTINUE
450         CONTINUE
460       CONTINUE
470     CONTINUE
        GOTO 599

C       OPTIMISATION3
C       =============

500     CONTINUE

        WRITE(*,*)'VOISINAGE='
        READ(*,*)NVOI

C       GENERATION D'UNE COMBINAISON

        DO 510 I=0,NFILT/2
        HO(I)=HA(I)
510     CONTINUE
        EO=EA

        DO 520 I=0,NFILT/2
        HC(I)=HA(I)
520     CONTINUE

        DO 570 I=0,NFILT/2
          DO 560 II=-NVOI,NVOI
            DO 550 J=I+1,NFILT/2
              DO 540 JJ=-NVOI,NVOI
                DO 535 K=J+1,NFILT/2
                  DO 530 KK=-NVOI,NVOI
                  HC(I)=HA(I)+II/BIT
                  HC(J)=HA(J)+JJ/BIT
                  HC(K)=HA(K)+KK/BIT

C       CALCUL DE L'ERREUR

        CALL ERR(NFILT,HC,NBAND,NFI,NFS,DN,WN,PI,EC)

C       TEST

        IF(EC.GT.EO) GOTO 530
        EO=EC
        HO(I)=HC(I)
        HO(J)=HC(J)
        HO(K)=HC(K)
530               CONTINUE
535             CONTINUE
540           CONTINUE
550         CONTINUE
```

```
560      CONTINUE
570    CONTINUE
       GOTO 599

599    CONTINUE
       DO 600 I=0,NFILT-1
600    WRITE(*,*)'HO(I)=',HO(I)
       WRITE(*,650)EI,EA,EO
650    FORMAT(3E15.8)

       GOTO 135


3000   CONTINUE
       WRITE(*,3100)
3100   FORMAT(////' 1-CHANGER LES DONNEES'/' 2-SORTIR'
      * //' CHOISIR UN NUMERO,')
       READ(*,*)IZ
       IF(IZ.EQ.1) GOTO 1
       STOP
       END

C      ================================================
       SUBROUTINE ERR(NFILT,HC,NBAND,NFI,NFS,DN,WN,PI,EC)
C      ================================================
C
C      Sous-programme de calcul de la norme de l'erreur maximale
C
       DIMENSION HC(64)
       DIMENSION DN(500),WN(500)
       INTEGER NFI(8),NFS(8)

       NF=NFILT/2
       EC=0.
       DO 5400 K=1,NBAND
         DO 5300 J=NFI(K),NFS(K)
         IF(NFILT.NE.2*NF) THEN
         HF=HC(NF)
           DO 5100 I=0,NF-1
           HF=HF+2*HC(I)*COS(PI*(NF-I)*J/NFS(NBAND))
5100       CONTINUE
         ELSE
         HF=0.
           DO 5200 I=0,NF
           HF=HF+2*HC(I)*COS(PI*(NF-I)*J/NFS(NBAND))
5200       CONTINUE
         ENDIF

         ER=ABS(WN(J)*(DN(J)-HF))

         IF(ER.LT.EC) GOTO 5300
         EC=ER
5300     CONTINUE
5400   CONTINUE
       RETURN
       END
```