Un Système d'Analyse et de Synthèse de Parole par Prédiction Linéaire pour un Taux de Transmission inférieur a 2400 BPS

Claude Side

82-02

INRS-Télécommunications
a/s Recherches Bell-Northern Ltée.
3, Place du Commerce
Ile des Soeurs, Qué.
H3E 1H6

Rapport Technique de l'INRS-Télécommunications No. 82-02

février 1982

UN SYSTEME D'ANALYSE ET DE SYNTHESE DE PAROLE PAR PREDICTION LINEAIRE POUR UN TAUX DE TRANSMISSION INFERIEUR A 2400 BPS

CLAUDE SIDE

Je remercie l'I.N.R.S.-Télécommunications de m'avoir acceuilli à Montréal et plus particulièrement, les professeurs M.Lennig et P.Kabal pour les conseils qu'ils m'ont donnés tout au long du déroulement de mon stage.

TABLE DES MATIERES

PRESENTATION

Ce projet a pour but la mise au point d'un système d'analyse et de synthèse de parole par une méthode de prédiction linéaire.Deux algorithmes distincts sont utilisés.Le premier, pour l'analyse, s'articule de la façon suivante:

- Détection de périodes 'pitch' par segmentation en zones voisées ou non.
- Lissage des périodes 'pitch'.
- Calcul, de façon adaptative, de la longueur de la structure d'analyse.
- Préemphasage.
- Fenêtrage.
- Calcul des coefficients d'autocorrélation.
- Calcul des coefficients de reflection du filtre de prédiction et du gain et leur quantification.
- Transmission à un taux inférieur à 2400 bits par seconde des paramètres d'analyse, c'est-à-dire la longueur de la structure, sa qualité (voisée,transition,non voisée), les paramètres de segmentation, les coefficients de réflection et le gain.

Le second algorithme effectue la synthèse à partir des paramètres qui ont été transmis. Pour cela, il calcule d'abord le résidu (ou erreur de prédiction) qui sera placé à l'entrée du filtre de prédiction. Le signal en sortie subit ensuite un désemphasage pour obtenir le signal synthétique final.

CHAPITRE I - QUELQUES RAPPELS SUR LA PREDICTION LINEAIRE

I-1 PRINCIPE

Considérons un signal de parole échantillonné à une certaine fréquence Fo (généralement égale à 8 KHz) et appelons s(i) les échantillons ainsi obtenus. La théorie consiste à exprimer la valeur du nième échantillon s(n) comme combinaison linéaire d'un nombre fini p d'échantillons précédents et d'une certaine entrée e(n) suivant la formule :

$$s(n) = -\sum_{K=1}^{P} a(k)s(n-k) + e(n)$$

En termes de transformée en Z, la relation précédente donne

$$S(z)(1 + \sum_{K=1}^{P} a(k)\ z^{-K}) = E(z)$$

En posant :

$$a(0) = 1 \quad \text{et} \quad A(z) = \sum_{K=0}^{P} a(k)\ z^{-K}$$

on obtient :

$$S(z) = E(z) \; / \; A(z)$$

Le signal S(z) apparait donc comme la réponse du filtre 1/A(z), ne comportant que des pôles, à l'entrée E(z). Différentes méthodes de calcul du filtre 1/A(z) sont connues -13, 14, 15, 23-. Elles ont toutes pour principe le calcul d'une certaine erreur de prédiction, fonction des coefficients a(k), et la minimisation de celle ci par un critère quadratique. L'erreur de prédiction, appelée aussi résidu, consiste en la différence entre l'échantillon de signal s(n) et son estimation par :

$$\tilde{s}(n) = -\sum_{k=1}^{p} a(k)\, s(n-k)$$

De cette façon, le résidu est :

$$e(n) = s(n) - \tilde{s}(n)$$

Donc :

$$e(n) = s(n) + \sum_{k=1}^{p} a(k)\, s(n-k)$$

On remarque ainsi que l'erreur de prédiction n'est autre que

l'entrée du filtre de prédiction considérée précédemment.

Le critère quadratique choisi ici est celui de la méthode par autocorrélation, et consiste en la minimisation de la quantité E suivante par rapport aux a(k) :

$$E = \sum_{-\infty}^{+\infty} e(n)^2$$

La sommation est, bien entendu, limitée à la fenêtre considérée et

$$\partial E / \partial a(k) = 0 \qquad k=1,\ldots,p$$

Le lecteur en trouvera toutes les indications dans les références (13, 14, 15, 23).

CHAPITRE II - DESCRIPTION DE L'ALGORITHME D'ANALYSE

L'algorithme d'analyse se fait d'après le synoptique de la figure 1 et son organigramme est décrit en annexe 1.

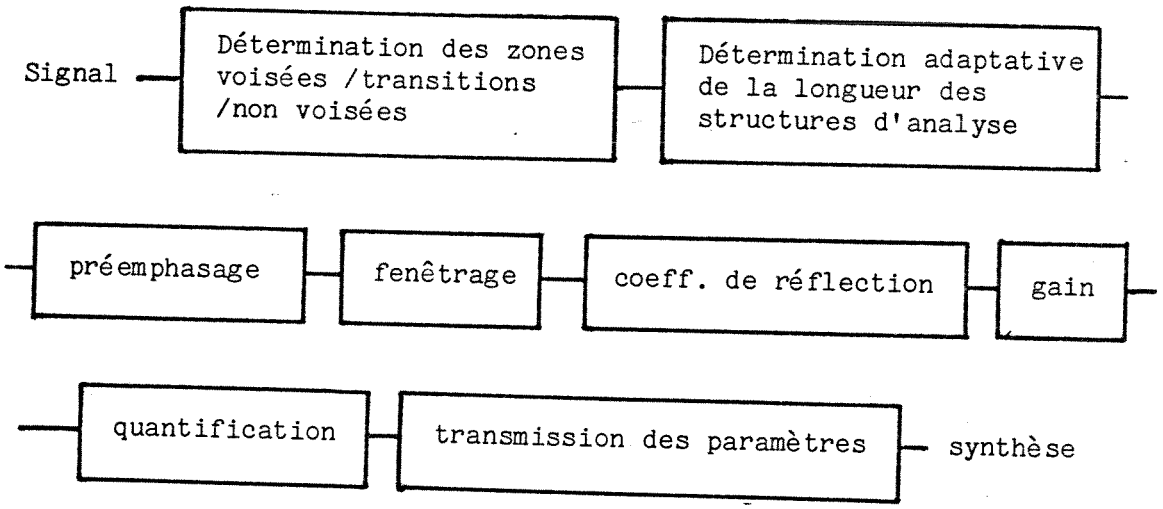


figure 1 - algorithme d'analyse

Dans la suite, nous allons voir ses différentes parties plus en détail.

II-1 ESTIMATION DES PERIODES PITCH

II-1-1 CHOIX DE LA METHODE DE DETECTION

Le choix s'est basé sur l'article référencé (1) et les

descriptions des différents algorithmes (2 à 8). Il s'est porté sur la méthode des detecteurs parallèles de Gold et Rabiner (2) qui, entre autres propriétés, effectue une analyse directe du signal de parole dans le domaine temporel et qui, du point de vue calculatoire, peut être utilisé en traitement parallèle, ce qui augmente sa rapidité d'éxécution.

II-1-2 PRINCIPE

L'analyse de Pitch comprend les différentes étapes (voir figure 2) :

- Filtrage du signal de parole
- Traitement du signal pour obtenir des trains d'impulsions élémentaires ayant la périodicité du signal.
- Estimation de la période pitch de chaque train d'impulsions par des détecteurs simples.
- Détection de la période finale à partir de combinaisons logiques de ces estimations.

Le but du filtrage est de sélectionner la région correspondante au premier formant et qui est à peu près celle de la fréquence fondamentale du signal. Le filtre utilisé est de type passe bas avec une fréquence de coupure à 400 Hz. On effectue un traitement du signal de façon à obtenir 6 trains d'impulsions élémentaires M1 à M6 (voir figure 3) :

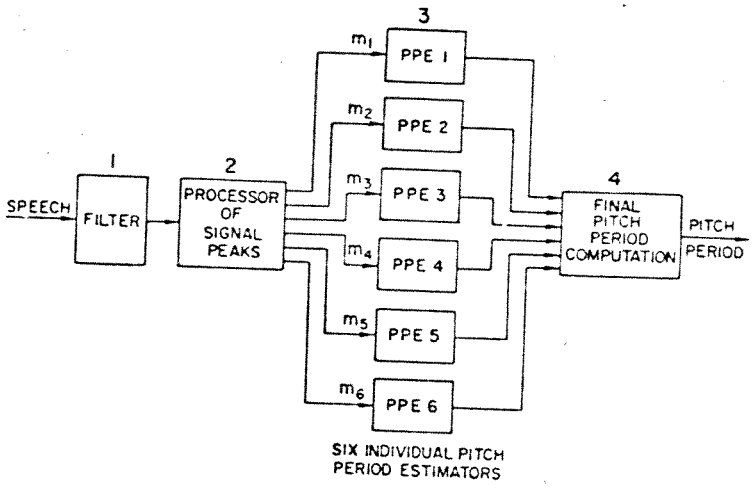


figure 2 - Principe de l'analyse de pitch par détecteurs parallèles (d'après 2)

M1 : impulsion située à chaque pic et ayant son amplitude

M2 : impulsion située à chaque pic et d'amplitude égale à la différence entre celle du pic et celle de la vallée précédente

M3 : impulsion située à chaque pic et d'amplitude égale à la différence entre celle du pic et celle du pic précédent (0 si la différence est négative)

M4 : impulsion située à chaque vallée et d'amplitude égale à l'opposé de celle de celle ci

M5 : impulsion située à chaque vallée et d'amplitude égale à la somme de celle du pic précédent et de l'opposé de celle de

la vallée

M6 : impulsion située à chaque vallée et d'amplitude égale à la somme de celle du minimum local précédent et de l'opposé de celle de la vallée (0 si elle est négative)

Chaque train d'impulsions est alors traité par un système non linéaire à temps variable et à décroissance exponentielle : un pic ayant été détecté, nous attendront un certain temps mort variable T suivi d'une décroissance exponentielle avant de détecter le suivant (voir figure 4). Le taux de décroissance exponentiel B et le temps mort T dépendent du plus récent estimé de la période pitch élémentaire, celle ci étant égale à la longueur entre deux impulsions. On a choisi (4):

$$B = Pav / 1.39$$

$$T = 0.4\ Pav$$

On limite d'autre part la période pitch entre les valeurs 0.4 ms et 10 ms.Ces période élémentaires étant ainsi définies, on forme une matrice 6x6 contenant ces estimés et des combinaisons linéaires de ceux-ci, puis, on calcule la période pitch finale par une méthode par coincidences. Une décision de non voisé est prise lorsque l'écart entre les périodes est supérieur à 15%.

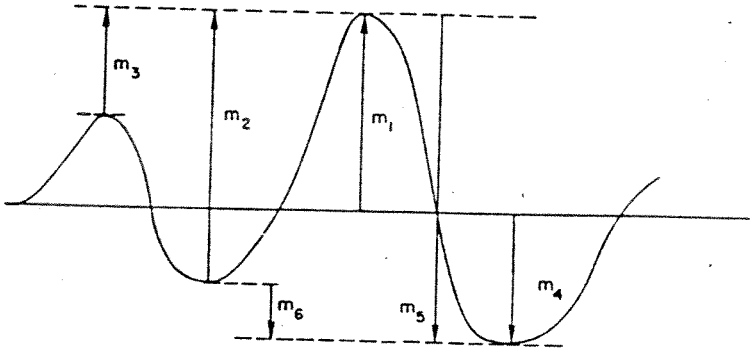


figure 3 - Détermination des trains d'impulsions élémentaires (extrait de 2)

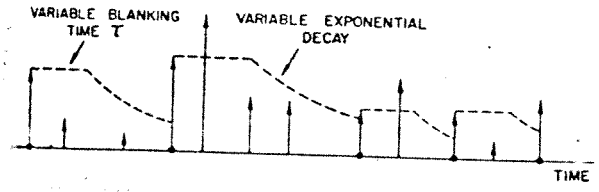


figure 4 - Fonctionnement des circuits de détection (extrait de 2)

II-1-3 ALGORITHME

L'algorithme utilise à la base le principe énoncé précédemment. Cependant, un critère de décision plus élaboré a été introduit pour ce qui est de la qualité de la section. Trois états sont possibles :

VOISE : les périodes se suivent avec un écart inférieur à 15%.

TRANSITION : chaque fois qu'un écart trop important entre les périodes est décelé; ceci coincide avec généralement au passage d'un état voisé à un état non voisé ou inversement.

NON VOISE : chaque fois qu'on ne peut déceler une succession de périodes.

Le programme ne fournit pas explicitement la valeur de la période mais donne en sortie la position en unité d'échantillon de la fin de la section (celle-ci étant égale au début de la section suivante) et sa qualité. Ceci correspond à l'option choisie pour le calcul du résidu, lors de la synthèse, puisqu'il consistera à envoyer une impulsion de Dirac à chaque début de section. La figure 5 montre un exemple de détection de différents segments sur un prélèvement de signal de parole prononcé par un locuteur mâle; V correspond à l'état voisé, T à une transition et U à un état non voisé.

II-1-4 LISSAGE

Afin de minimiser au maximum les erreurs concernant les

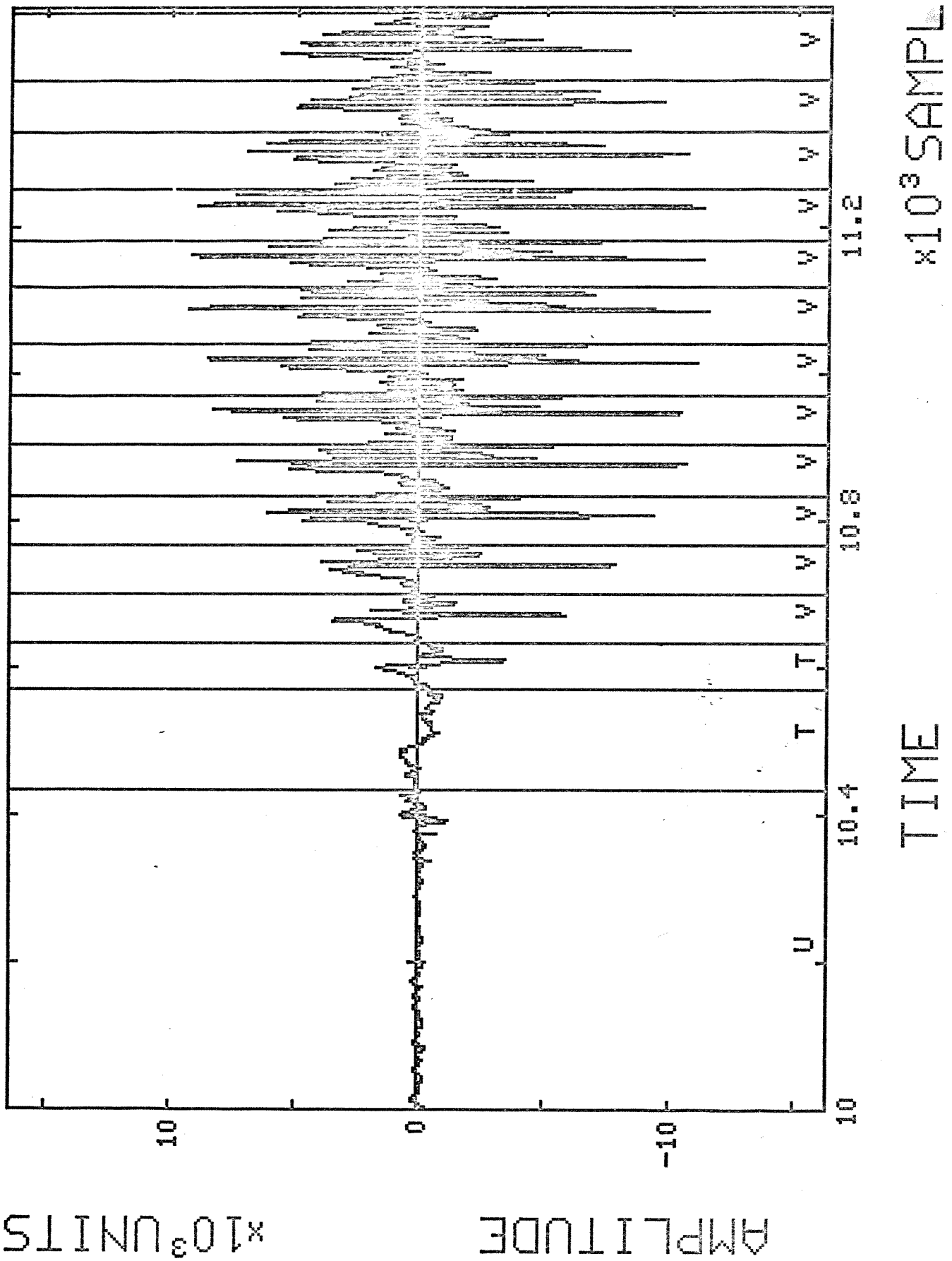


figure 5 - exemple de segmentation d'un signal audio

périodes pitch, un algorithme de lissage a été introduit. Il isole d'abord les sections comprises entre deux segments non voisées, puis effectue une vérification des décisions prises de façon à éliminer un contexte voisé-transition-voisé pour trois sections successives et à le remplacer par un contexte voisé-voisé-voisé. L'étape suivante consiste à faire un lissage des fréquences fondamentales de chaque section par une médiane d'ordre 3 puis par une médiane d'ordre 5. Ceci permet,en particulier, d'éliminer les erreurs de décision telle que la détection d'une section de transition à l'intérieur d'une zone fortement voisée et a aussi pour effet d'améliorer le naturel de la voix obtenue par synthèse.

II-2 DETERMINATION DE LA LONGUEUR D'UNE STRUCTURE D'ANALYSE

Pour éviter qu'une structure ne débute ou ne se termine au beau milieu d'une section, ce qui entrainerait une dégradation de la parole synthétique, il a été choisi d'en déterminer sa longueur de façon adaptative. Ainsi, la structure ne pourra contenir qu'un nombre entier de sections, celles-ci étant toutes du même type et dans le cas voisé,de périodes pitch proches l'une de l'autre (avec une tolérance de 15%). De plus, l'analyse sera faite dés que la structure contiendra un nombre d'échantillons aussi proche que possible d'une limite fixée, tout en lui étant inférieur. Cette limite est prise égale à 256 échantillons dans le cas voisé (32 ms pour une fréquence d'échantillonnage de 8

KHz) et à 512 échantillons dans les deux autres cas (62.5 ms). Si la section contient plus d'échantillons que le nombre limite,comme c'est souvent le cas pour un segment non voisé, nous diviserons la section en un nombre entier de sous sections et nous ferons l'analyse sur chacune d'elles.

II-3 PREEMPHASAGE ET FENETRAGE

On effectue un préemphasage (13) en faisant passer le signal à travers un filtre d'ordre 1 de fonction de transfert $1-(MU)z^{-1}$ Le coefficient de préemphasage MU est compris entre 0.9 et 1. Son choix est adaptatif (13) et il se calcule suivant la formule:

$$MU = R(1) / R(0)$$

R(0) et R(1) étant les deux premiers coefficients d'autocorrélation du signal.

Si ce calcul n'est pas possible (par exemple en cas de silence), la valeur par défaut est 0.9375.

En ce qui concerne le fenêtrage, et afin d'éliminer les effets de bord et de réduire les distorsions spectrales, le choix s'est porté sur une fenêtre de type Hamming qui permet une meilleure atténuation à l'extérieur de la bande passante par comparaison à une fenêtre rectangulaire. Le fenêtrage satisfait

l'équation suivante :

$w(n) = (1 - a) - a.\cos(2 \Pi (n-1)/(N-1))$ avec $a = 0.46$

N étant le nombre total d'échantillons de la structure.

II-4 CALCUL DES COEFFICIENTS DE REFLECTION ET DU GAIN

Le calcul des coefficients de réflection s'effectue en résolvant un système d'équations de Toeplitz (13,14,15,23). La résolution utilise une forme modifiée de l'algorithme de Durbin proposée par J.Leroux et C.J.Gueguen (16) et dont l'algorithme se trouve à la figure 6. Nous calculeront 10 coefficients dans le cas d'une structure voisée et seulement 4 coefficients dans les autres cas.

Le calcul du gain se fait en utilisant la formule :

$$G = \sqrt{R(0) \prod_{i=1}^{P} (1 - K(i)^2)}$$

R(0) étant le premier coefficient d'autocorrélation et les K(i) étant les coefficients de réflection.

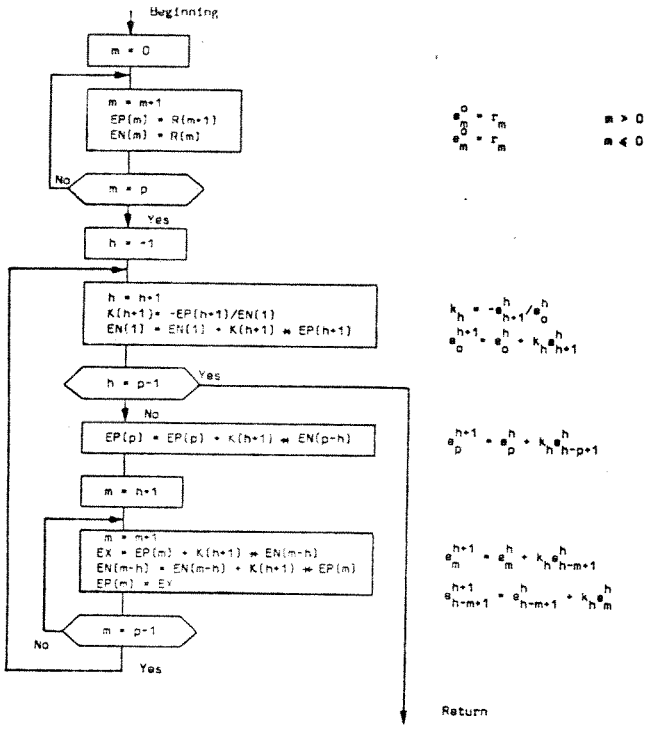


figure 6 - Calcul des coefficients de réflection (extrait de I6)

CHAPITRE III - QUANTIFICATION ET TRANSMISSION

III-1 CAS DES COEFFICIENTS DE REFLECTION

Le nombre de niveaux alloués à chaque coefficients de réflection obéit au tableau suivant :

K(1) à K(4)	32 niveaux	5 bits chacun
K(5) à K(8)	16 niveaux	4 bits chacun
K(9)	8 niveaux	3 bits
K(10)	4 niveaux	2 bits

- Le principe de quantification de ces coefficients s'inspire d'une méthode proposée dans la référence (22) et utilisée, dans le cadre d'une implémentation en temps réél, pour une adaptation du système en nombres entiers.

Pour les deux premiers coefficients, on effectue la quantification linéaire de la quantité $\mathrm{Log}(1-K(i))/(1+K(i))$ (optimale selon Makhoul (15)). L'importance des autres coefficients étant moindre, nous en effectuerons une quantification linéaire. Le principe consiste à multiplier chaque coefficient de réflection par 2^{14} de façon à le placer sur une échelle de 15 bits (niveaux allant de -2^{14} à 2^{14})lui ajouter un biais, de multiplier le résultat par un facteur d'échelle pour

se ramener entre les valeurs +64 et -64, puis de quantifier linéairement sur 128 niveaux et tronquer le résultat compte tenu du nombre de bits autorisé pour la transmission. Il a été démontré que les deux dernières opérations reviennent au même que de quantifier linéairement entre +64 et -64 sur une échelle dont le nombre de niveaux est celui affecté à chaque coefficient.

Une étude effectuée à l'I.N.R.S. (18) a montré que les coefficients de réflection évoluent en majorité entre les valeurs minimales et maximales de la figure 7. Appelant Vmin(i) et Vmax(i) les valeurs minimale et maximale de chaque coefficient K(i), le biais B(i) et le facteur d'échelle S(i) vérifient le système d'équations suivant :

$$\begin{cases} -64 = (-2^{14}\ x\mathrm{Vmin}(i) + B(i))xS(i) \\ 64 = (\ 2^{14}\ x\mathrm{Vmax}(i) + B(i))xS(i) \end{cases}$$

Sa résolution donne pour le biais et le facteur d'échelle les valeurs se trouvant à la figure 8. Rappelons que, pour chaque coefficient K(i), on quantifie la quantité Q(i) entre -64 et 64 :

$$Q(i) = (K(i)x2^{14} + B(i))xS(i)$$

A la réception, nous effectuerons l'opération inverse pour récupérer le coefficient de réflection K(i); c'est à dire :

$$K(i) = (Q(i) / S(i) - B(i)) / 2^{14}$$

III-2 QUANTIFICATION ET TRANSMISSION DES AUTRES PARAMETRES

a) Le gain est quantifié suivant une échelle pseudo logarithmique sur 32 niveaux (5 bits) et avec des valeurs maximum et minimum respectivement égales à 100000 et 10.

b) La longueur d'une structure peut atteindre jusqu'à 512 points dans le cas non voisé, nous le quantifierons donc sur 9 bits.

c) Le nombre de sections par structure sera quantifié sur 4 bits.

d) La qualité de la structure pouvant prendre 3 états, sera quantifiée sur 2 bits.

e) Le coefficient de préemphasage sera quantifié sur 5 bits avec des valeurs minimum et maximum de 0.9 et 1 . Il faut noter que l'on peut, si l'on veut économiser le nombre de bits, choisir un coefficient de préemphasage fixe égal à la valeur par défaut.

f) Le nombre total de bits nécessaire à la transmission est donc pour une structure voisée de :

$$25 + 41 = 66 \text{ bits}$$

et pour une structure non voisée ou de transition de :

25 + 20 = 45 bits

On tombe ainsi bien en dessous du taux de 2400 bits par seconde. En effet, même si toutes les structures étaient voisées, on aurait, pour une fréquence d échantillonnage de 8 KHz, 32 structures dans une seconde, et donc, 2112 bits.

coeff. de réflection	Vmin	Vmax
K(1)	-0.9728	0.7677
K(2)	-0.3721	0.8902
K(3)	-0.8184	0.5677
K(4)	-0.3006	0.7647
K(5)	-0.4668	0.3822
K(6)	-0.2022	0.6353
K(7)	-0.2586	0.4079
K(8)	-0.1651	0.5806
K(9)	-0.418	0.512
K(10)	-0.5	0.266

figure 7 - Valeurs minimum et maximum des coefficients de réflection

coeff. de réflection	Biais	Facteur d'échelle
K(3)	2048.	0.0056
K(4)	-3850.	0.0073
K(5)	696.	0.0091
K(6)	-3522.	0.0092
K(7)	-1229.	0.0117
K(8)	-3400.	0.0103
K(9)	-768.	0.0084
K(10)	1920.	0.0102

figure 8 - biais et facteurs d'échelle

CHAPITRE IV - DESCRIPTION DE L'ALGORITHME DE SYNTHESE

L'algorithme de synthèse se fait d'après le synoptique de la figure 9.

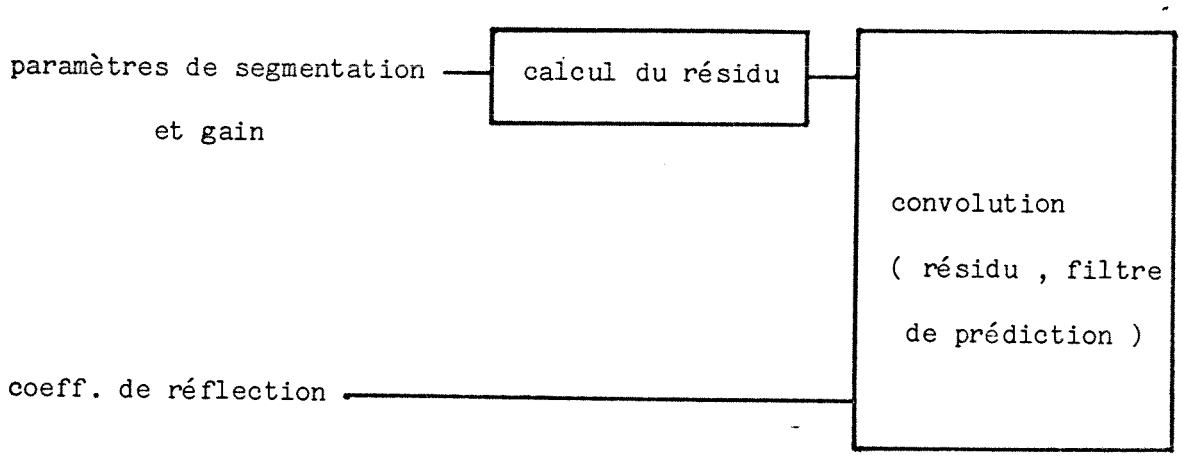


figure 9 - algorithme de synthèse

Rappelons que les paramètres d'entrée sont les coefficients de réflection, le gain, la longueur et le nombre de sections de la structure, sa qualité, le coefficient de préemphasage. Son organigramme se trouve en annexe 1.

IV-1 CALCUL DU RESIDU

Le calcul du résidu diffère suivant le type de structure envisagée.

a) Cas d'une structure voisée

Le résidu consiste en une suite d'impulsions d'amplitude égale au gain et situées à chaque début de section. Une approximation est faite ici qui consiste à considérer que les sections d'une même structure sont toutes de longueur identique.

b) Cas d'une structure non voisée

Le résidu est alors un bruit blanc Gaussien centré suivant une loi normale N(0,G), G étant le gain de la structure.

c) Cas d'une transition

Le résidu est dans ce cas une hybridation des deux cas précédents; c'est à dire qu'on enverra une impulsion d'amplitude G/2 au début de la structure et un bruit blanc Gaussien centré de loi normale N(0,G/2). D'autres valeurs ont été essayées pour le facteur de multiplication à affecter au gain; mais c'est 1/2 qui semble donner les meilleurs résultats.

IV-2 SYNTHESE DU SIGNAL

La synthèse s'effectue en plaçant le résidu calculé précédemment à l'entrée d'un filtre en treillis obtenu à partir des coefficients de réflection.

Un problème est apparu à ce niveau, en ce sens que la structure contenant le signal synthétisé présentait une assez grande différence d'énergie avec la structure de signal original. Ce problème a été résolu en multipliant le gain par un facteur adéquat et en recalculant alors le résidu et le signal synthétique. Ce facteur a été pris égal au rapport de l'énergie de la structure de signal original Eo par celle de la structure de signal synthétique Es ou ce qui revient au même à la quantité:

$$G \;/\; Es\sqrt{\prod_{i=1}^{P}(1-K(i)^2)}$$

G étant le gain de la structure analysée.

CHAPITRE V - CONSIDERATIONS SUR LA MISE EN OEUVRE EN TEMPS REEL

Le programme principal prélève des tampons successifs d'un fichier audio en vue d'en faire l'analyse par segmentation, l'analyse par prédiction linéaire puis d'effectuer une synthèse. Entre chaque tampon, le programme conserve en mémoire la position du dernier segment du tampon précédent et les valeurs des échantillons compris entre ce dernier segment et la fin du tampon car ceux ci n'ont pas étés utilisés. La longueur du tampon a été fixée à 500 échantillons, on effectue de plus un chevauchement de 60 échantillons,qui est l'ordre du filtre utilisé pour l'analyse par segmentation, par conséquent, le retard dù à la seule mise en mémoire des échantillons du tampon est au minimum de 70 ms pour une fréquence d'échantillonage de 8 Khz. Si l'on tient compte des échantillons non utilisés du tampon précédent, on arrive généralement à un retard de l'ordre de 85 ms. Il faudrait y ajouter le temps effectif de calcul pour obtenir le retard total entre le prélévement du tampon et la production des échantillons synthétisés.

CHAPITRE VI - RESULTATS ET CONCLUSION

Les figures 10,11,12 montrent les différents résultats obtenus. La figure 10 illustre la méthode de reconstruction du signal telle qu'elle a été décrite précédemment (impulsion dans le cas voisé, bruit blanc dans le cas non voisé et combinaison des deux dans le cas transition). Les figures 11 et 12 montrent la comparaison entre le signal original et le signal synthétisé; la phrase illustrée ici est prononcée par un locuteur mâle et a pour contenu: Add the sum to the product of these three . On note que le signal synthétisé suit d'assez près le signal original aussi bien du point de vue temporel (figure 11) que du point de vue fréquentiel (figure 12).

Du point de vue auditif, le signal reconstruit est compréhensible et intelligible aussi bien dans le cas d'un locuteur que d'une locutrice, la voix se rapproche assez bien de la voix originale dans le cas mâle tandis que pour une voix femelle,elle est sensiblement plus rauque (pertes de hautes fréquences).

Ce projet consistait à modéliser un système utilisant la prédiction linéaire. Il a montré, toutefois, que l'application pure et simple de la théorie était insuffisante pour obtenir un signal totalement identique au signal original. Ainsi, certaines investigations, en ce qui concerne la forme du résidu (21) et l'utilisation du gain comme amplitude des impulsions, pourraient être d'un intérêt certain.

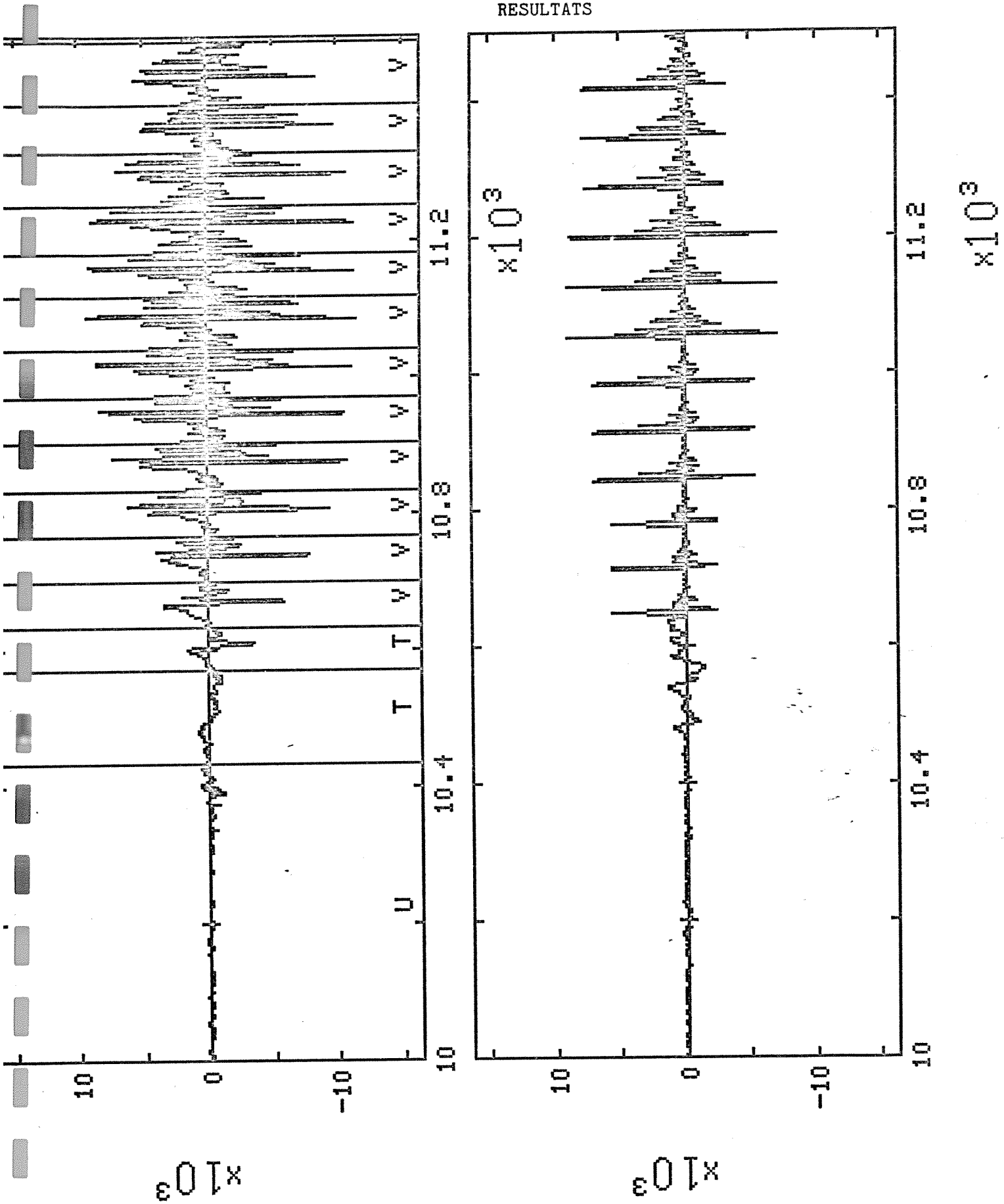


figure 10 - Construction du signal synthétisé

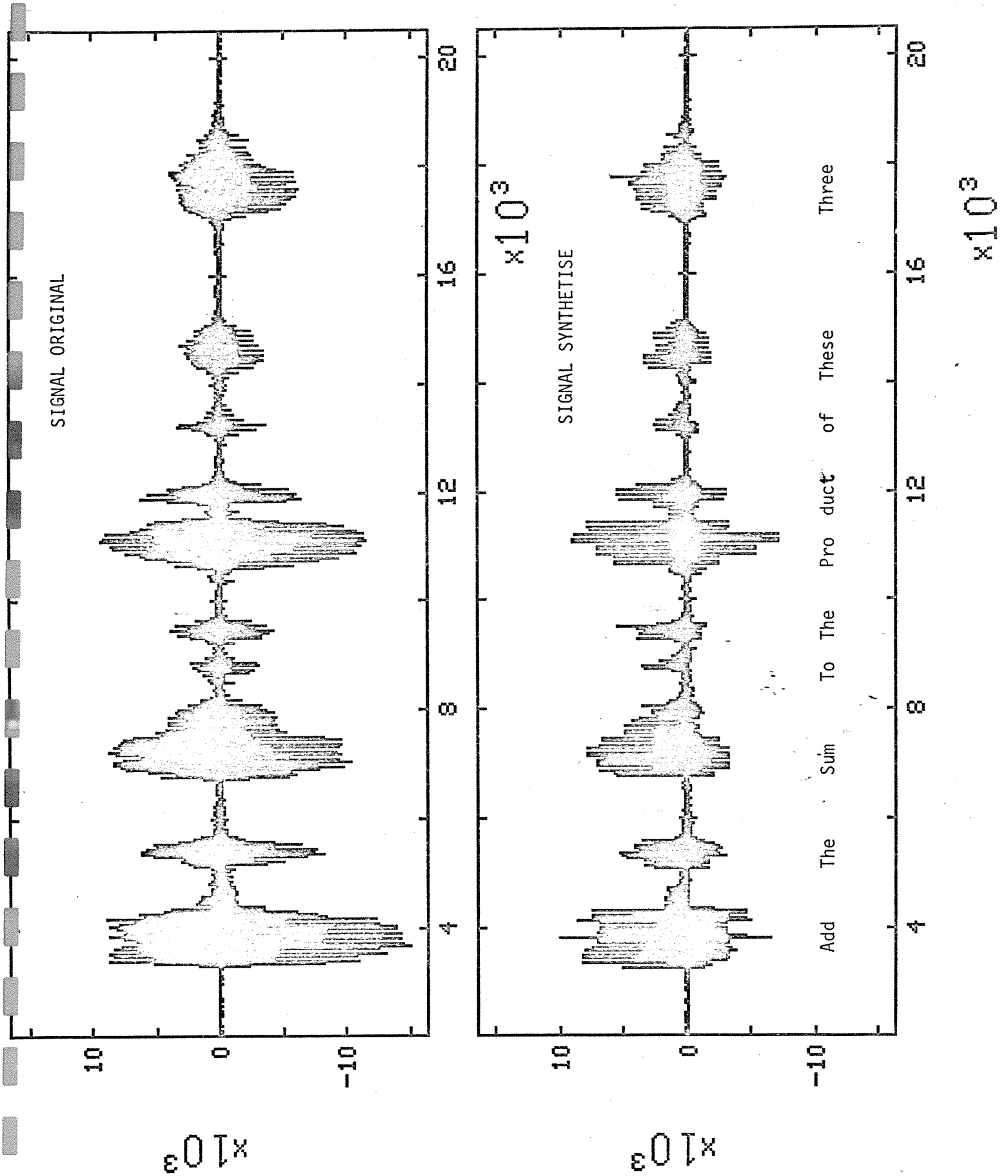


figure II - Comparaison entre le signal original et le signal synthétique - Domaine temporel

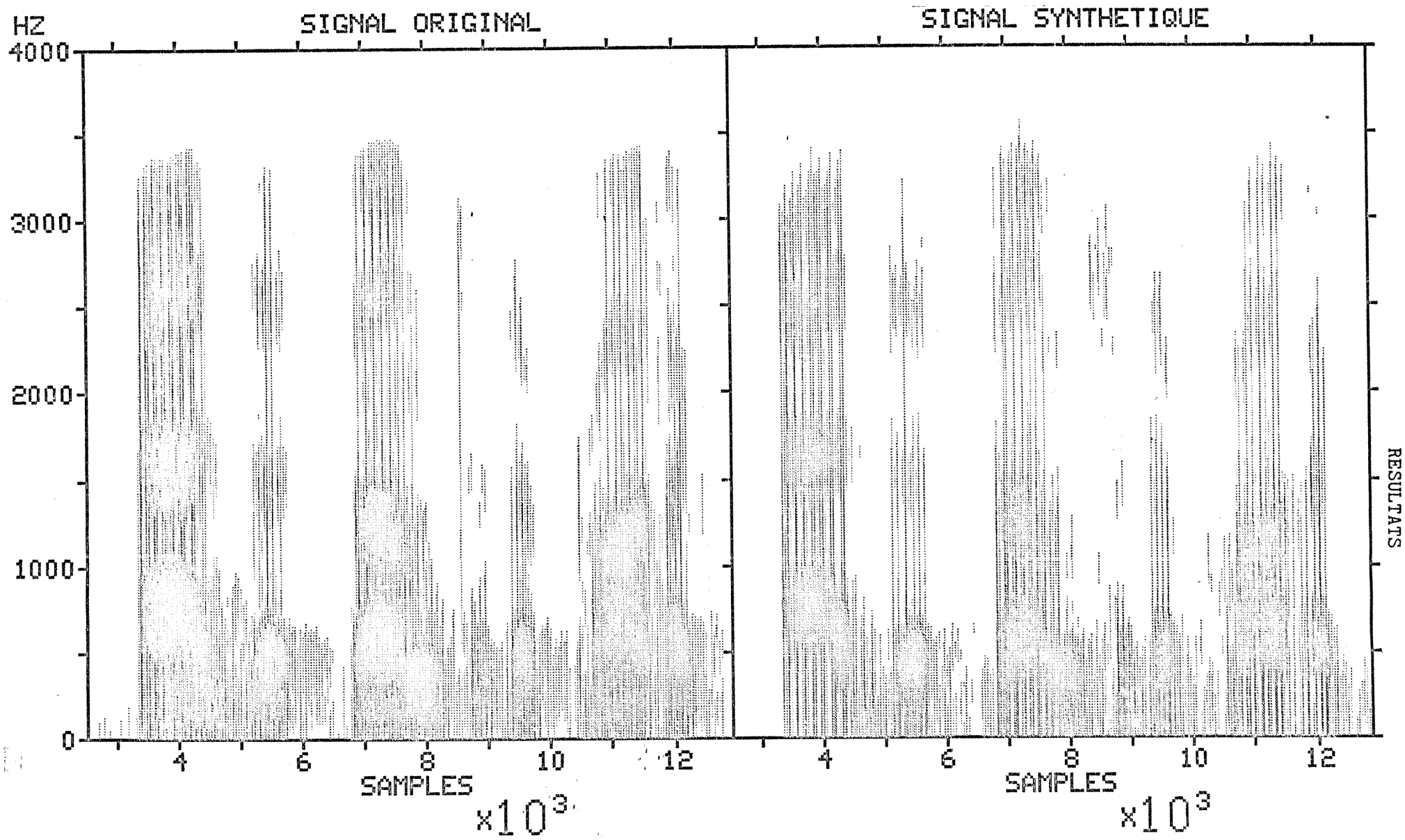


figure I2 - Comparaison entre le signal original et le signal synthetique
Domaine frequentiel

BIBLIOGRAPHIE

1- A comparative performance study of several pitch detection algorithms - L.R.Rabiner, M.J.Cheng, A.E.Rosenberg, C.A.McGonegal - IEEE Transactions on Acoustics, Speech, and Signal Proceesing, Vol.ASSP-24, NO.5, October 76

2- Parallel processing techniques for estimating pitch periods of speech in the time domain - B.Gold, L.Rabiner - The Journal of the Acoustical Society of America, Vol.46, pages 442-448, 1969

3- The Sift Algorithm for Fundamental Frequency Estimation - J.D.Markel - IEEE Transactions on Audio and Electroacoustics, Vol.AU-20, NO.5, December 72

4- Average Magnitude Difference Function Pitch Extractor - M.J.Ross, H.L.Shaffer, A.Cohen, R.Freudberg, H.J.Manlev - IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-22, NO.5, October 74

5- On the use of Autocorelation Analysis for Pitch Detection - L.R.Rabiner - IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-25, NO.1, February 77

6- Maximum Likelihood Pitch Estimation - J.D.Wise, J.R.Caprio, T.W.Parks - IEEE Transactions on Acoustics, Speech,and Signal Processing, Vol.ASSP-24, NO.5, October 76

7- Pseudo Maximum Likelihood Speech Pitch Extraction - D.H.Friedman - IEEE Transactions on Acoustics, Speech and Signal Processing, Vol.ASSP-25, NO.3, June 77

8- Real-Time Digital Hardware Pitch Detector - J.J.Dubnowski,

BIBLIOGRAPHIE

R.W.Schafer, L.R.Rabiner - IEEE TRansactions on Acoustics, Speech, and Signal Processing - Vol.ASSP-24, NO.1, February 76

9- Microprocessor Realization of a Linear Predictive Vocoder - E.M.Hofstetter, J.Tierney, O.Wheeler - IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-25, NO.5, October 77

10- Optimum Classification of Voiced Speech, Unvoiced Speech, and Silence in the Presence of Noise and Interference - R.J.McAulay - Technical Note 1976-7, 3 June 76, Lincoln Laboratory, M.I.T.

11- Robust Speech Processing - B.Gold - Technical Note 1976-6, 27 January 76, Lincoln Laboratory, M.I.T.

12- Design of a Robust Maximum Likelihood Pitch Estimator for Speech in Additive Noise - R.J.McAulay - Technical Note 1979-28, 11 June 79, Lincoln Laboratory, M.I.T.

13- Linear Prediction of Speech - J.Markel, A.H.Gray - Springer-Verlag 76

14- Digital Processing of Speech Signal - L.R.Rabiner, R.W.Schafer - Prentice Hall 78

15- Linear Prediction. A Tutorial Review - IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-63, April 75

16- A Fixed Point Computation of Partial Correlation Coefficients - J.Leroux, C.Gueguen - IEEE Transactions on Acoustics, Speech, and Signal Processing, June 1977, p. 257-259

17- Les Processus de la Communication Parlée - J.S.Lienard

18- Feasibility Study of a Hardware Implementation of a 4.8 kb/s RELP Speech Coder - P.Kabal - INRS -Telecommunications Technical Report No.81-08, May 81

19- An Improved Linear Predictive Coding Algorithm for Speech Transmission at 2.4 kb/s - D.C.Stevenson - Bell-Northern Research Unpublished Technical Memorandum, December 80

20- Comparative Evaluation of Residual-Excited Linear Prediction and Sub-Band Coding for Speech Transmission at 9.6 kb/s - P.kabal, D.C.Stevenson - Bell-Northern Research Unpublished Technical Memorandum,October 79

21- Quality Limitations of LPC Speech Coding Techniques - D.C.Stevenson - Bell-Northern Research Unpublished Working Paper, July 80

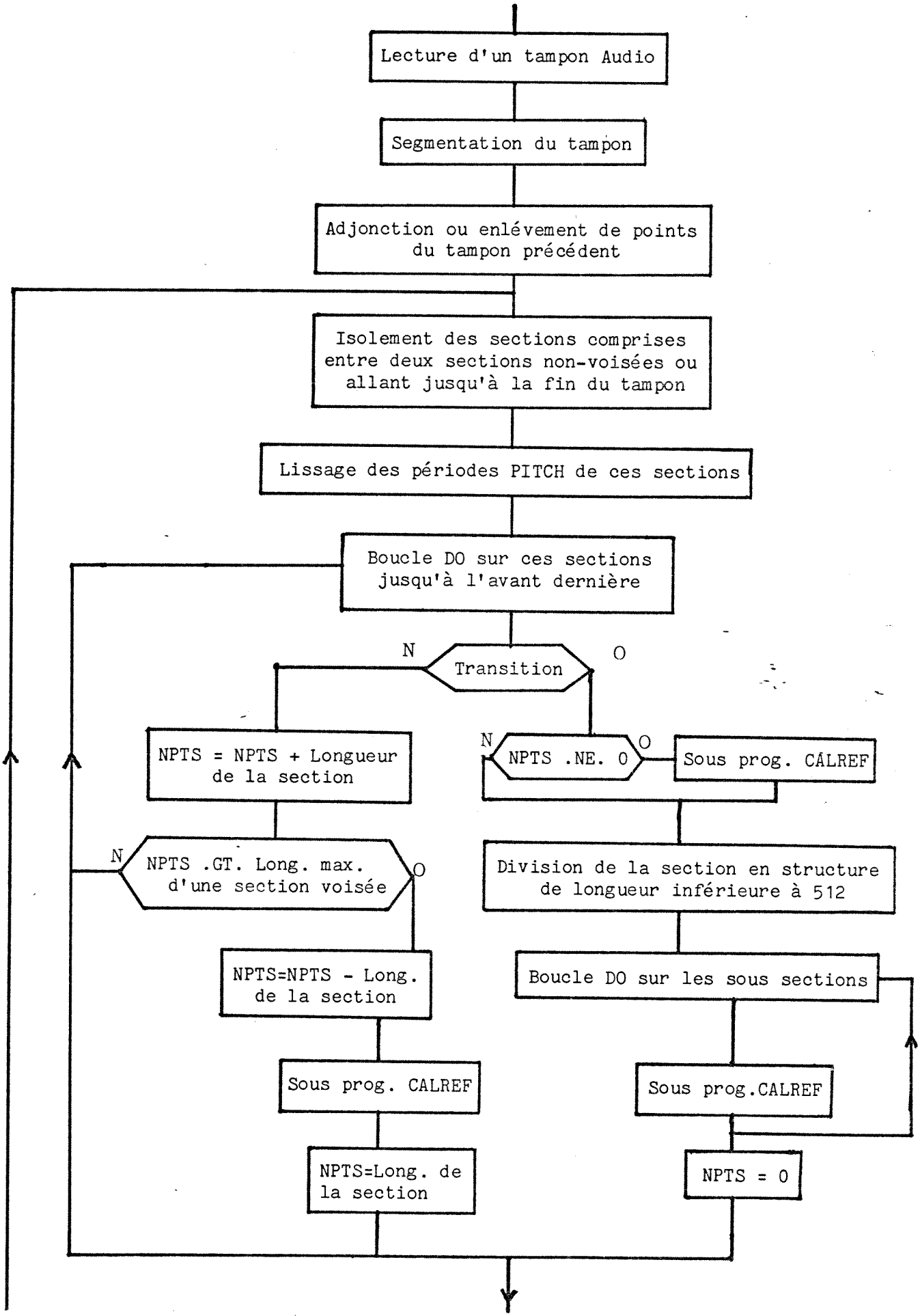
22- Communication Personnelle du Centre de Recherche - Non Publié

23- Analyse et Synthèse de Parole par Prédiction Linéaire - C.Side, M.Lasry - Projet à l'Ecole Nationale Supérieure des Télécommunications de Paris, Juin 81

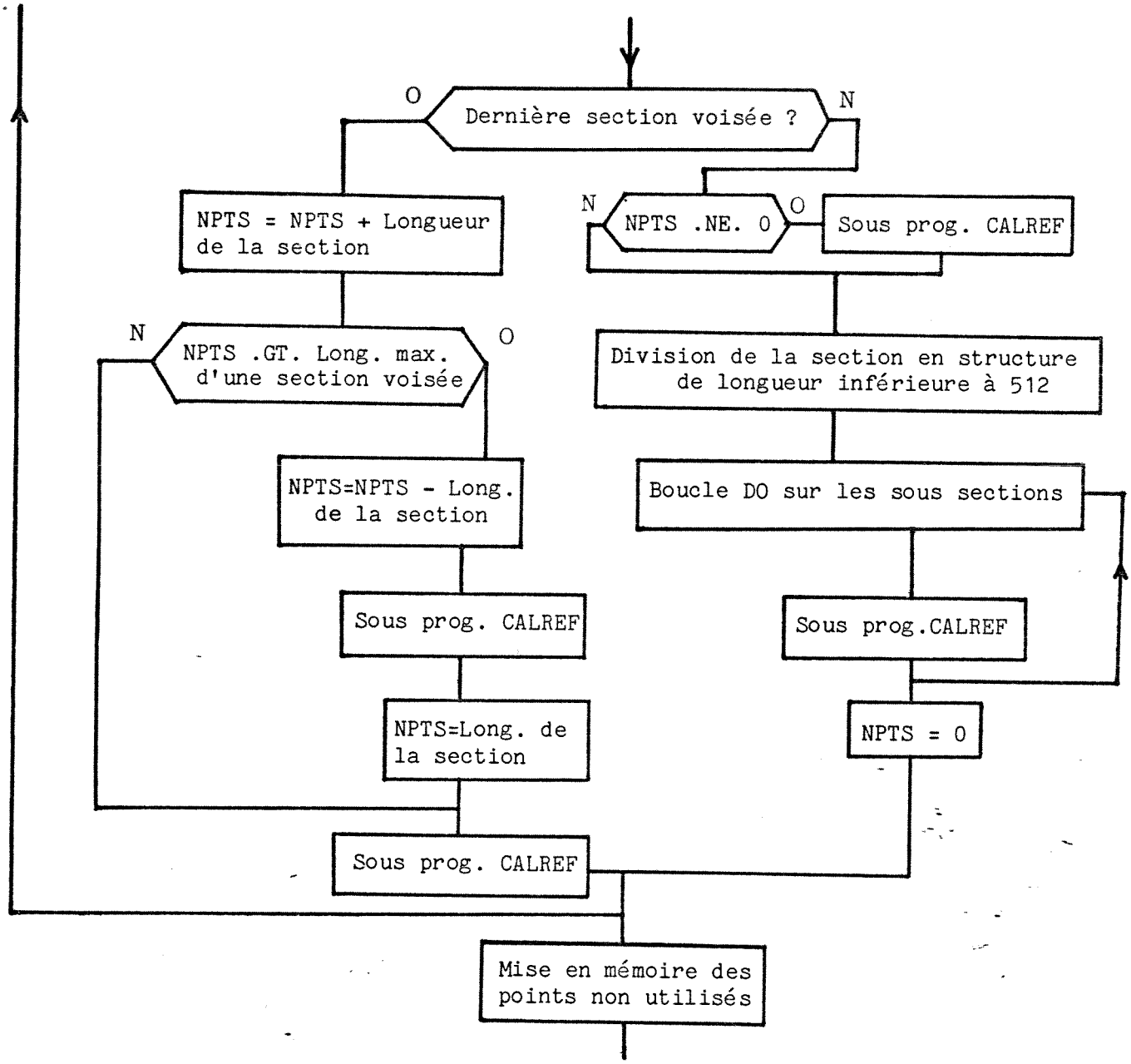
- ANNEXE 1 -

ORGANIGRAMMES DU PROGRAMME ET DES PRINCIPAUX SOUS-PROGRAMMES

PROGRAMME PRINCIPAL ANSYL

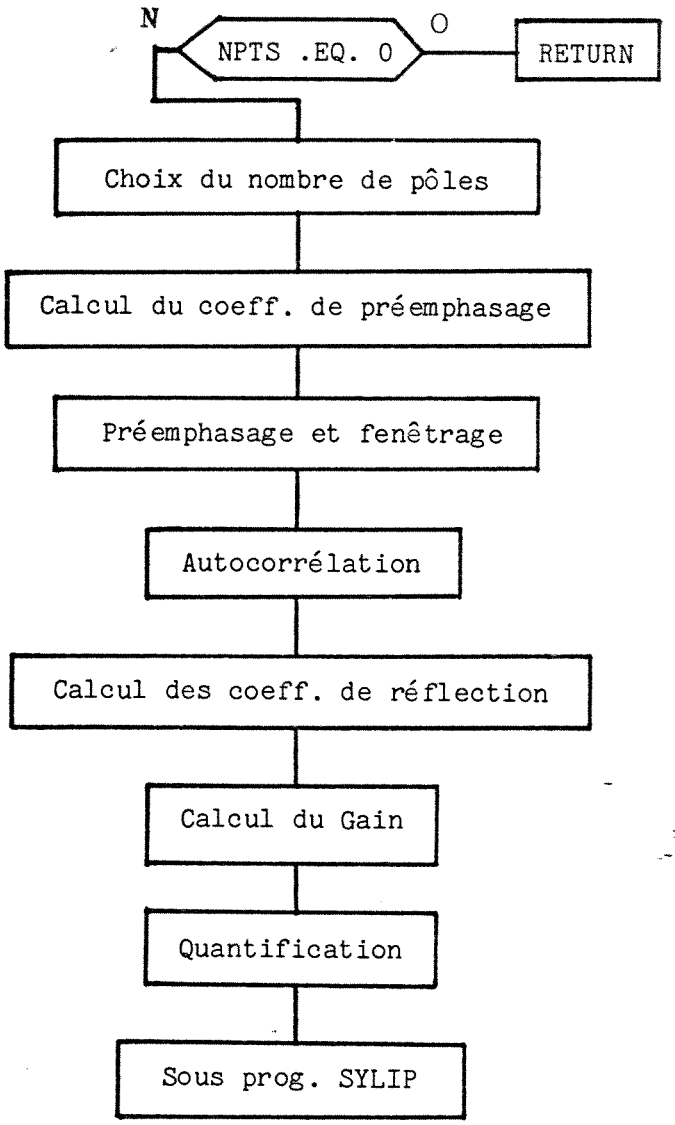


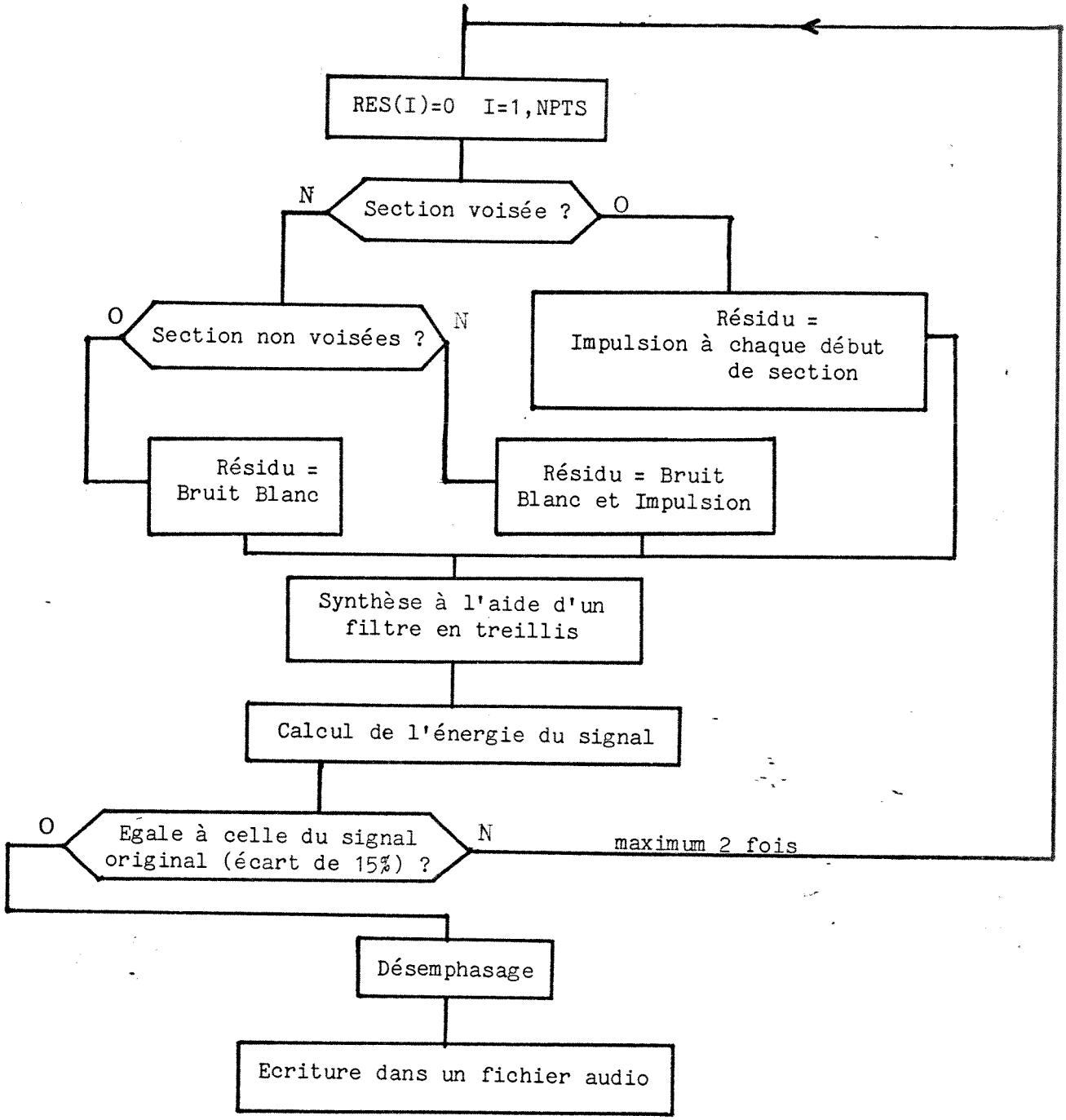
PROGRAMME PRINCIPAL ANSYL



REMARQUE : La dernière partie du programme pourrait être traitée directement dans la boucle DO sur les sections . Cependant , L'organigramme permet de traiter séparément les sections de transition et donc d'en changer facilement les paramètres tels que la longueur maximale de la structure .

SOUS PROGRAMME CALREF




RES(I)=0 I=1,NPTS
N
Section voisée ?
O
O
Section non voisées ?
N
Résidu = Impulsion à chaque début de section
Résidu = Bruit Blanc
Résidu = Bruit Blanc et Impulsion
Synthèse à l'aide d'un filtre en treillis
Calcul de l'énergie du signal
O
Egale à celle du signal original (écart de 15%) ?
N
maximum 2 fois
Désemphasage
Ecriture dans un fichier audio

- ANNEXE 2 -

PROGRAMMES FORTRAN

LISTE DES PROGRAMMES

ANSYL - Programme principal

CALREF - Analyse par prédiction linéaire

PITCHLISS - Lissage des périodes 'pitch'

SYLIP - Synthèse par prédiction linéaire

SEGBIS - Détermination des périodes 'pitch'

PPROC - Extraction des périodes d'un tampon audio

ACORR - Calcul des coefficients d'autocorrélation

AUTOK - Calcul des coefficients de réflection

DEEM - Désemphasage du signal

FILTRE - Filtrage d'un tampon audio (utilisé par PPROC)

ISHIFT - Décalement des éléments d'un vecteur

LATSYN - Synthése du signal à l'aide d'un filtre en treillis

LISS - Lissage d'un tableau de nombres entiers

PREEM - Préemphasage du signal

QNTDFN, QNTLAR, QNTLOG, QUANTZ - Quantification

RANGE - Rangement de tableaux

WNDGEN - Fenêtrage

```
C----------Bell-Northern Research / INRS Computer Laboratory----------
C
C
C MODULE: PREDICTION LINEAIRE     ANSYL
C                                 CALREF
C                                 PITCHLISS
C
C FILE NAME: ANSYL.FOR
C
C
C PURPOSE: ANALYSE  ET SYNTHESE PAR PREDICTION LINEAIRE
C
C
C DESCRIPTION: ANSYL FAIT D'ABORD APPEL AU SOUS PROGRAMME SEGBIS QUI
C              DETERMINE LES SECTIONS VOISEES/TRANSITIONS/NON VOISEES DU
C              FICHIER AUDIO.IL CALCULE ENSUITE LA LONGUEUR D'UNE STRUCTURE
C              D'ANALYSE DE FACON ADAPTATIVE ET FAIT ALORS APPEL A CALREF.
C
C              PITCHLISS EFFECTUE UN LISSAGE DES FREQUENCES DE PITCH .
C
C              CALREF CALCULE LES COEFFICIENTS DE REFLECTION POUR UNE STRUCTURE
C              DONNEE ET FAIT ENSUITE APPEL AU SOUS PROGRAMME SYLIP POUR
C              SYNTHETISER LE SIGNAL
C
C              ON EFFECTUE UN CHEVAUCHEMENT DE 10% DE PART ET D'AUTRE DE LA
C              STRUCTURE D'ANALYSE.
C
C              LE COEFFICIENT DE PREEMPHASAGE EST CALCULE DE FACON ADAPTATIVE.
C
C PARAMETERS:
C       FILNAM - NOM DU FICHIER AUDIO ( DE TYPE '[AUDIO.NATURAL].AUD')
C       FNAME - NOM DU FICHIER CONTENANT LE SIGNAL SYNTHETISE (DE TYPE
C                '[SIDE.AUDI].REL')
C
C       IBUFF - TAMPON AUDIO CONTENANT LES ECHANTILLONS DE SIGNAL
C       IBUFL - NOMBRE D'ECHANTILLONS CONSIDERES
C       LV - TABLEAU CONTENANT LES POSITIONS DES FINS DE SECTIONS
C       IPR - TABLEAU DONNANT LES QUALITES DES SECTIONS :
C               '0' DANS LE CAS VOISE
C               '1' DANS LE CAS TRANSITION
C               '2' DANS LE CAS NON-VOISE
C       IFO - TABLEAU CONTENANT LES FREQUENCES PITCH DES SECTIONS
C
C       IFRM - NUMERO DE LA STRUCTURE
C       LENV - LONGUEUR D'UNE STRUCTURE VOISEE
C       LENU - LONGUEUR D'UNE STRUCTURE NON VOISEE
C       NLEV - NOMBRE DE NIVEAUX DE QUANTIFICATION
C    (*)NPOV - NOMBRE DE POLES DU MODELE D'ANALYSE POUR UNE STRUCTURE VOISEE
C    (*)NPOU - NOMBRE DE POLES POUR UNE STRUCTURE NON VOISEE
C    (*)NPTS - NOMBRE D'ECHANTILLONS DU SIGNAL DANS UNE STRUCTURE
C    (*)PRE - FACTEUR DE PREEMPHASAGE
C       RBUFF - TABLEAU DES ECHANTILLONS BRUTS DE SIGNAL
C       REFL - TABLEAU CONTENANT LES COEFFICIENTS DE REFLECTION
C    (*)REFQ - TABLEAU CONTENANT LES COEFFICIENTS DE REFLECTION
C              APRES QUANTIFICATION
C       RXX - TABLEAU CONTENANT LES COEFFICIENTS D'AUTOCORRELATION
C       SIG - TABLEAU CONTENANT LES ECHANTILLONS DE SIGNAL APRES PREEMPHASAGE
C       WND - COEFFICIENTS DE FENETRAGE (WND=0.46 POUR UNE FENETRE DE
C             HAMMING)
C       XQ - TABLEAU CONTENANT LES POINTS POUR LA QUANTIFICATION
C            DES COEFFICIENTS DE REFLECTION
```



```
C       YQ - TABLEAU CONTENANT LES NIVEAUX DE SORTIE DE QUANTIFICATION
C
C ROUTINES REQUIRED:
C       ACORR - CALCULE L'AUTOCORRELATION D'UN VECTEUR
C       AUTOR - CALCULE LES COEFFICIENTS DE REFLECTION
C       CLOSEFC - REFERME UN FICHIER AUDIO
C       DEFNAM - DEFINIT LE NOM COMPLET DU FICHIER
C       GETCML - DEFINIT LE NOM D'UN FICHIER
C       GETIPT - LIT UN NOMBRE DE POINTS EN ENTREE
C       ISHIFT - DECALE LES ELEMENTS D'UN TABLEAU
C       OPENRD - OUVRE UN FICHIER AUDIO POUR LECTURE
C       OPENWI - OUVRE UN FICHIER AUDIO POUR ECRITURE
C       PUTRPT - ECRIT LES ECHANTILLONS DANS UN FICHIER AUDIO
C       SEGBIS - SEGMENTE LE FICHIER AUDIO EN SECTIONS VOISEES/NON VOISEES
C
C            SEGBIS UTILISE LES SOUS PROGRAMMES PPROC,INPROC,RANGE,INFIL,
C            FILTRE ET LES DEUX FICHIERS DE COEFFICIENTS D10C61 ET D8C61.
C
C       SYLIP - SYNTHETISE LE SIGNAL A PARTIR DES PARAMETRES DE PREDICTION
C       PREEM - PREEMPHASE LE SIGNAL
C       QNTLAR - CALCULE LES NIVEAUX DE QUANTIFICATION
C       QNTUFM - CALCULE LES NIVEAUX DE QUANTIFICATION DE FACON UNIFORME
C       QUANTZ - QUANTIFIE LE SIGNAL
C       WNDGEN - FENETRAGE DU SIGNAL
C
C AUTHOR / MAINTAINED BY: C. SIDE
C
C
C DATE CREATED: 81/11/02
C
C
C UPDATES:
C
C
C----------Bell-Northern Research / INRS Computer Laboratory----------



      PARAMETER LUN1=2,LUN2=3,LUN3=4,IPRT=-1
      PARAMETER LENV=256,LENU=512,IBUFL=500,LMAX=100,TOL=0.15

      INTEGER PNTLUS
      INTEGER LV(LMAX),IFO(LMAX),IPR(LMAX)
      INTEGER IBUFF(IBUFL+IBUFL+LMAX),IMEM(IBUFL)
      CHARACTER*80 FILNAM,FNAME,FILCOF
C
C   LECTURE DU NOM DU FICHIER
C
C
      CALL GETCML('$FICHIER AUDIO:',FNAME,NCHR,ISRC)
      IF(ISRC.LE.0.OR.NCHR.LE.0)STOP 'AUCUN FICHIER'

C
C   OUVERTURE DES FICHIERS
C
      CALL DEFNAM(FNAME,FILNAM,NCHR,'[AUDIO.NATURAL].AUD')
      CALL OPENRD(LUN1,FILNAM,NSAMP,SFREQ,IPRT,ISTAT)
      IF(ISTAT.NE.0) GOTO 100

      CALL DEFNAM(FNAME,FNAME,NCHR,'(SIDE.AUDI].REL')
      CALL OPENWI(LUN3,FNAME,NSAMP,SFREQ,IPRT,ISTAT)
```

```
      IF(ISTAT.NE.0) GO TO 100

C
C     INITIALISATION
C

C     Initialisation du sous programme SEGBIS

      CALL INSEG
      CALL IFZ(LUN1)

      IFRM=0
      J=0
      NPTS=0
      LE=1
      LV(1)=0
      IFO(1)=0
      IPR(1)=2

C
C     Ouverture du fichier de coefficients du filtre FIR.
C

      IF(SFREQ.EQ.10000.)FILCOF='(SIDE.LIPRE)D10C61'
      IF(SFREQ.EQ.8000.) FILCOF='(SIDE.LIPRE)D8C61'

C     Initialisation du sous programme FILIRE

      CALL INFIL(FILCOF,NTAP)

      IF (NTAP.LE.0) STOP 'ERREUR: COEFFICIENTS - SEGBIS'
      NS2 = NTAP/2
      MEM=NS2
      NADV=IBUFL
      LEFRM=IBUFL+NS2

C     Initialisation du sous programme PPROC

      CALL INPROC (SFREQ,0,1)

      PNTLUS = -MEM

      DO WHILE(PNTLUS.LT.NSAMP)
C
C     LECTURE D'UN TAMPON AUDIO
C

        CALL GETIPT(LUN1,PNTLUS,IBUFF,LEFRM,NOUT)
        IF(NOUT.LE. 0) STOP 'ERREUR LECTURE'

C
C     SEGMENTATION DU TAMPON AUDIO EN ZONES VOISEES/NON VOISEES
C

      IOFFS=0
      LP=1
      LV(1)=LV(LE)
      IFO(1)=IFO(LE)
      IPR(1)=IPR(LE)
      LE=1

      CALL SEGBIS (IBUFF,NOUT,PNTLUS,NS2,SFREQ,LE,LV,IFO,IPR)
```



```
      IF(LV(1).LE.PNTLUS) THEN
       CALL ISHIFT(IBUFF,NOUT,LV(1)-PNTLUS)
       DO I=1,PNTLUS-LV(1)
        IBUFF(I)=IMEM(I)
       END DO
       NOUT=NOUT+PNTLUS-LV(1)
      ELSE
       NOUT=NOUT+PNTLUS-LV(1)
       CALL ISHIFT(IBUFF,NOUT,LV(1)-PNTLUS)
      END IF

      PNTLUS=PNTLUS+NADV

      IF(PNTLUS.GE.NSAMP.AND.LV(LE).GE.(NSAMP-NS2)) LV(LE)=NSAMP

C     ISOLEMENT DES SECTIONS COMPRISES ENTRE DEUX SECTIONS NON VOISEES OU
C     ALLANT JUSQU'A LA FIN DU TAMPON

      L=0
200   L=L+1
      IF(LP+L.LT.LE.AND.IPR(LP+L).NE.2) GO TO 200

C
C     CALCUL DE LA LONGUEUR DE LA STRUCTURE . CELLE CI EST AUSSI PROCHE QUE
C     POSSIBLE DE LENV POUR UNE STRUCTURE VOISEE ET DE LENU POUR UNE STRUCTURE
C     NON VOISEE .
C

      IF(L.GT.1)THEN

      CALL PITCHLISS(L+1,LV(LP),IFO(LP),IPR(LP),SFREQ)

       DO 50 I=LP+1,LP+L-1

       IF(IPR(I).EQ.1)THEN

        IF(NPTS.NE.0)CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,
     S   NPTS,0,J)

        N=(LV(I)-LV(I-1))/LENU+1
        NPTS=(LV(I)-LV(I-1))/N
        J=1

        IF(N.GT.1)THEN
         DO 45 K=1,N-1
45       CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS,IPR(I),J)
         NPTS=LV(I)-LV(I-1)-(N-1)*NPTS
        END IF

        CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS,IPR(I),J)
        NPTS=0
        J=0

       ELSE

        NPTS=NPTS+LV(I)-LV(I-1)
        J=J+1

        IF((ABS(IFO(I)-IFO(I-1)).GT.TOL*IFO(I-1)
```

```
     S       .OR.NPTS.GT.LENV).AND.(LV(I)-LV(I-1)).NE.NPTS)THEN
             CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS-(LV(I)-LV(I-1))
     S                        ,0,J-1)
             NPTS=LV(I)-LV(I-1)
             J=1
            END IF

           END IF

50         CONTINUE

          END IF

          IF(IPR(LP+L).EQ.0) THEN
           NPTS=NPTS+LV(LP+L)-LV(LP+L-1)
           J=J+1
           IF(NPTS.GT.LENV.AND.(LV(LP+L)-LV(LP+L-1)).NE.NPTS) THEN
            CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,
     S        NPTS-(LV(LP+L)-LV(LP+L-1)),0,J-1)
            NPTS=LV(LP+L)-LV(LP+L-1)
            J=1
           END IF

           CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS,0,J)
           NPTS=0
           J=0

          ELSE

           IF(NPTS.NE.0)THEN
            CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS,0,J)
            NPTS=0
            J=0
           END IF

           N=(LV(LP+L)-LV(LP+L-1))/LENU+1
           NPTS=(LV(LP+L)-LV(LP+L-1))/N
           J=1

            IF(N.GT.1)THEN
             DO 40 I=1,N-1
40           CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS,IPR(LP+L),J)
             NPTS=LV(LP+L)-LV(LP+L-1)-(N-1)*NPTS
           END IF

           CALL CALREF(IBUFF,NOUT,IOFFS,LUN3,IFRM,NPTS,IPR(LP+L),J)
           NPTS=0
           J=0
          END IF

          LP=LP+L
          L=0
          IF(LP.LT.LE) GO TO 200

          IF(LV(LE).LT.PNTLUS) THEN
           DO I=1,PNTLUS-LV(LE)
            IFRM(I)=IBUFF(NOUT-NS2-(PNTLUS-LV(LE))+I)
           END DO
          END IF

         END DO
```



```
          CALL FDUMPI(LUN3,0.,0)
100       CALL CLOSEFL(LUN1)
          CALL CLOSEFL(LUN3)
          STOP
          END

C
C    SOUS PROGRAMME D'ANALISE PAR PREDICTION LINEAIRE
C
          SUBROUTINE CALREF(IBUFF,NOUT,IOFFS,LUN,IFRM,NPTS,IPR,N)

          PARAMETER WND=0.46 ,NPOMAX=10 ,NPMAX=512,LMAX=100
          PARAMETER NLEMAX=32,NPOV=10,APOO=4,ALPH=0.1
          REAL XO(NLEMAX),YO(NLEMAX)
          REAL REFL(NPOMAX),REFO(NPOMAX),RXX(NPOMAX+1)
          REAL RBUFF(NPMAX+LMAX),SIG(NPMAX+LMAX),WIV(NPMAX+LMAX)
          REAL BIAS(NPOMAX),SCA(NPOMAX)
          INTEGER NLEV(NPOMAX),IBUFF(NOUT)


          DATA BIAS/0.,0.,2048.,-3850.,696.,-3522.,-1229.,-3400,-768.,1920./
          DATA SCA/1.,1.,.0056,.0073,.0091,.0092,.0117,.0103,.0084,.0102/
          DATA NLEV/32,32,32,32,16,16,16,16,8,4/

          IF(NPTS.EQ.0) RETURN

          PRE=0.9375
          AMEM=0.

          IFRM=IFRM+1

          NPOLE=NPOV
          IF(IPR.EQ.0) NPOLE=NPOV

          IDEB=IOFFS-ALPH*NPTS
          IF(IDEB.LE.0) IDEB=1
          IFIN=IOFFS+NPTS+ALPH*NPTS
          IF(IFIN.GT.NOUT) IFIN=NOUT

          DO I=1,IFIN-IDEB
           RBUFF(I)=IBUFF(IDEB+I)
          END DO
          LEN=IFIN-IDEB

          IOFFS=IOFFS+NPTS

C
C    CALCUL DU COEFFICIENT DE PREEMPHASAGE OPTIMUM
C
          R0=0.
          R1=0.
          DO 30 I=1,LEN-1
          R0=R0+RBUFF(I)**2
30        R1=R1+RBUFF(I)*RBUFF(I+1)
          R0=R0+RBUFF(LEN)**2
          IF(R0.GT.0.)PRE=R1/R0
C
C    PREEMPHASAGE ET FENETRAGE
C
```

```
      CALL PREEM(MBUFF,SIG,LEN,PRE,AMEM)
      CALL MNOGEN(MIN,LEN,MND)
      DO 10 I=1,LEN
10    SIG(I)=SIG(I)*MIN(I)
C
C   ANALYSE
C
      CALL ACORR(SIG,LEN,RXX,NPOLE+1)
      CALL AUTOK(RXX,NPOLE,REFL)

C
C   CALCUL DU GAIN
C
      G=RXX(1)
      DO 70 I=1,NPOLE
70    G=G*(1-REFL(I)**2)
      G=SQRT(G)
C
C  QUANTIFICATION
C

      CALL GNTLAR(NLEV(1),-0.98,0.77,XQ,YQ)
      CALL QUANTZ(REFL(1),L,XQ,NLEV(1))
      REFQ(1)=YQ(L)

      CALL GNTLAR(NLEV(2),-0.38,0.69,XQ,YQ)
      CALL QUANTZ(REFL(2),L,XQ,NLEV(2))
      REFQ(2)=YQ(L)

      DO 20 I=3,NPOLE
      REFL(I)=(REFL(I) * 2.**14 + BIAS(I))*SCA(I)
      CALL GNIDFN(XQ,YQ,NLEV(I),'UNIFORM',64.)
      CALL QUANTZ(REFL(I),L,XQ,NLEV(I))
      REFQ(I)=YQ(L)
20    REFQ(I)=(REFQ(I)/SCA(I)-BIAS(I))/2.**14

      CALL GNTLOG(32,100.,10.,10000.,XQ,YQ)
      CALL QUANTZ(G,L,XQ,32)
      GQ=YQ(L)
C
C   APPEL DU SOUS PROGRAMME DE SYNTHESE
C

300   CALL SYLIP(LUN,IPR,PRE,REFQ,GQ,NPTS,N)

      RETURN
      END

C
C   SOUS PROGRAMME DE LISSAGE
C

      SUBROUTINE PITCHLISS(L,LV,IFQ,IPR,SFREQ)

      PARAMETER TOL=0.15
      INTEGER LV(L),IFQ(L),IPR(L)

      DO I=2,L-1
       IF(IPR(I-1).EQ.0.AND.IPR(I+1).EQ.0.AND.IPR(I).EQ.1) THEN
        IPR(I)=0
        IFQ(I)=SFREQ/(LV(I)-LV(I-1))
```



```
       END IF
      END DO

      IF(IFQ(L).NE.0.AND.IFQ(L-1).NE.0.AND.
     S   ABS(IFQ(L)-IFQ(L-1)).GT.TOL*IFQ(L-1))
     S   IFQ(L)= (IFQ(L)+IFQ(L-1))/2

      IF(IPR(1).NE.2) THEN
       CALL LISS(IFQ,L)
      ELSE
       CALL LISS(IFQ(2),L-1)
      END IF

      DO I=2,L
       IF(IFQ(I).NE.0) THEN
        LV(I)=LV(I-1)+ SFREQ/IFQ(I)
        IPR(I)=0
       ELSE
        IF(LV(I-1).GT.LV(I)) THEN
         IF(IPR(I).EQ.2) THEN
          LV(I-1)=LV(I)
         ELSE
          LV(I)=LV(I-1)
          IPR(I)=IPR(I-1)
         END IF
        END IF
       END IF
      END DO

      RETURN
       END
```

```
C
C
C----------Bell-Northern Research / INRS Computer Laboratory----------

      SUBROUTINE SILIP(LUN,IPR,PRE,REFL,G,NPTS,NS)

      PARAMETER NPMAX=512,NPOMAX=10,NPOV=10,NPOO=4,WND=0.46
      REAL REFL(NPOMAX),RES(NPMAX),SIG(NPMAX)
      REAL MEMEN(NPOMAX)
      DOUBLE PRECISION DSEED
      LOGICAL VERIF

C
C   INITIALISATION
C
      GI=G
      VERIF=.FALSE.
      ICOUNT=0

      DSEED=123457.D0

      DO 60 I=1,NPOLE
60    MEMEN(I)=0.
      AMEM=0.

      NPOLE=NPOO
      IF(IPR.EQ.0) NPOLE=NPOV

100   DO 40 I=1,NPTS
40    RES(I)=0.

C
C   CALCUL DU RESIDU
C
      IF(IPR.EQ.0)THEN
       NUM=0
       IF(NS.GT.1)THEN

       M=NPTS/NS
       DO 80 J=1,NS-1
       RES(NUM+1)=G
80     NUM=NUM+M
       END IF

       RES(NUM+1)=G

      ELSE

       IF(IPR.EQ.2) THEN

        CALL BRUIT(DSEED,NPTS,RES,G)

       ELSE

        CALL BRUIT (DSEED,NPTS,RES,G/2.)
        RES(1)=RES(1)+G/2.

       ENDIF
      END IF
```



```
----------Bell-Northern Research / INRS Computer Laboratory----------

MODULE: SUBROUTINE   SILIP (LUN,IPR,PRE,REFL,G,NPTS,NS)
                     BRUIT

FILE NAME: SILIP.FOR

PURPOSE: SYNTHESE PAR PREDICTION LINEAIRE

DESCRIPTION: CE PROGRAMME SYNTHETISE LE SIGNAL A PARTIR DES COEFFICIENTS DE
             REFLECTION ET DES SEGMENTATIONS VOISEES/TRANSITIONS/NON VOISEES
             DU FICHIER AUDIO.

             LE CALCUL DU RESIDU SE FAIT COMME SUIT:

              POUR UNE STRUCTURE VOISEE , UNE IMPULSION AU DEBUT DE CHAQUE
              SECTION D'AMPLITUDE EGALE AU GAIN .
              DANS LE CAS TRANSITION , MELANGE D'UNE IMPULSION D'AMPLITUDE
              EGALE A LA MOITIE DU GAIN ET D'UN BRUIT BLANC D'ENERGIE
              EGALE A LA MOITIE DU GAIN.
              DANS LE CAS NON VOISE , UN BRUIT BLANC D'ENERGIE EGALE AU GAIN.

             BRUIT EST UN SOUS PROGRAMME GENERANT UN BRUIT BLANC GAUSSIEN
             CENTRE

PARAMETERS:
      LUN = UNITE D'ECRITURE DES ECHANTILLONS DE SIGNAL SYNTHETISES
      G = GAIN DE LA STRUCTURE
      IPR = QUALITE DE LA SECTION:
                ='0' POUR UNE SECTION VOISEE
                ='1' POUR UNE TRANSITION
                ='2' POUR UNE SECTION NON VOISEE
      NPOLE = NOMBRE DE POLES DU FILTRE D'ANALYSE
      NPOV = NOMBRE DE POLES POUR UNE STRUCTURE VOISEE
      NPOO = NOMBRE DE POLES POUR UNE STRUCTURE NON VOISEE
      NPTS = NOMBRE D'ECHANTILLONS DANS LA STRUCTURE
      NS = NOMBRE DE SECTIONS DANS LA STRUCTURE
      PRE = COEFFICIENT DE DESEMPHASAGE
      REFL = TABLEAU CONTENANT LES COEFFICIENTS DE REFLECTION
      RES = TABLEAU CONTENANT LES VALEURS DU RESIDU
    (*)SIG = TABLEAU CONTENANT LES VALEURS DU SIGNAL

ROUTINES REQUIRED:
      DEEM = DESEMPHASE LE SIGNAL
      GGNRL = GENERE UN TABLEAU DE VALEURS ALEATOIRES SUIVANT UNE LOI N(0,1)
      LATSIN = SYNTHETISE LE SIGNAL
      POIEPI = ECRIT DANS UN FICHIER AUDIO

AUTHOR / MAINTAINED BY: C.SIDE

DATE CREATED: 81/11/02

UPDATES:
```

```
C
C  SYNTHESE
C
      CALL DATSYN (NPOLE,REFL,RES,NPTS,NCMEM,SIG)

      EN=0.
      DO I=1,NPTS
       EN=EN+SIG(I)**2
      END DO

      DO I=1,NPOLE
       EN=EN*(1-REFL(I)**2)
      END DO

      EN=SQRT(EN)

      IF(IPR.EQ.0.AND.ICOUNT.LT.2.AND.ABS(EN-GI).GT.0.15*GI) THEN
       G=G*GI/EN
       ICOUNT=ICOUNT+1
       VERIF=.TRUE.
      ELSE
       VERIF=.FALSE.
      END IF

      IF(VERIF) GO TO 100

  DESEMPHASAGE

      CALL DEEM(SIG,SIG,NPTS,PRE,AMEM)


  ECRITURE

      CALL POINPT(LUN,SIG,NPTS)

      RETURN
      END

SOUS PROGRAMME DE GENERATION D'UN BRUIT BLANC

      SUBROUTINE BRUIT(DSEED,NPOINTS,E,G)
      DIMENSION E(NPOINTS)
      DOUBLE PRECISION DSEED

      CALL GGNML(DSEED,NPOINTS,E)

      DIS=0.
      DO 20 I=1,NPOINTS
      DIS=DIS+E(I)**2
      DIS=SQRT(DIS)
      DO 10 I=1,NPOINTS
      E(I)=E(I)*G/DIS

      RETURN
      END
```

```
C--------------Bell-Northern Research / INRS Computer Laboratory-------------
C
C MODULE:       SEGBIS (IBUFFO,NOUT,PNTLUS,NS,SFREQ,N,LO,IFO,IPR)
C                ENTRY INSEG
C                INTEGER FUNCTION FINDZ(DEB,INC,LIM)
C                   ENTRY IFZ(LON)
C                INTEGER FUNCTION PNT(PAUD)
C
C FILE NAME:    SEGBIS.FOR
C
C PURPOSE:      Segmenter un tampon audio en zones voisees et non-voisees
C               et determiner les periodes individuelles pour les zones
C               voisees.
C               Ce programme s'inspire du programme PERLI de R. MARCHAND
C
C DESCRIPTION: SEGBIS utilise la sous-routine PPROC pour determiner les
C               periodes du fichier audio.
C
C PARAMETERS:   IBUFFO : Tampon audio.
C
C               NOUT : Dimension de IBUFF
C
C               PNTLUS : Numero du premier echantillon de IBUFFO dans le
C                        fichier audio .
C
C               NS : Longueur de chevauchement .
C
C               SFREQ : Frequence d'echantillonnage .
C
C               PARAMETRES DE SORTIE DANS L'ORDRE:
C
C                  N  : Dimension des tableaux LO,IFO,IPR
C
C                  LO : Tableau des fins de periode au point le plus proche
C                       de zero( CORRESPONDANT AU DEBUT DE LA PERIODE SUIVANTE)
C
C                  IFO : Tableau donnant les frequences des periodes .
C
C                  IPR : Caractere qui indique l'etat de la periode :
C                              '0'  :  periode voisee
C                              '1'  :  periode de transition
C                              '2'  :  periode non-voisee
C
C ROUTINES REQUIRED:  PPROC,RANGE,FILTRE
C
C AUTHOR / MAINTAINED BY:    C. SIDE
C
C DATE CREATED:        12 janvier 1982
C
C UPDATES:
C
C--------------Bell-Northern Research / INRS Computer Laboratory-------------
C
      SUBROUTINE SEGBIS(IBUFFO,NOUT,PNTLUS,NS,SFREQ,N,LO,IFO,IPR)

      IMPLICIT INTEGER (A-Z)
      PARAMETER MDIM=20,MAXTAP=255,BUFL=500,NMAX=100
      COMMON/PARAM/ TRIG,PMIN,PMAX,RUNDO,DLVEL,CLOCK,SIL,TAU,LEVEL
```



```
C
      LOGICAL VOISE,FIN1,TRANS,INIT
      REAL SFREQ,TOL,TAU,RUNDO,TEMPS
      DIMENSION WAVS(MDIM+1),IBUFFO(NOUT),IBUFF(BUFL+NMAX)
      DIMENSION P(BUFL),C(BUFL),CI(NMAX),CO(NMAX),Iw(NMAX),Lw(NMAX)
      DIMENSION LO(NMAX),IFO(NMAX),IPR(NMAX)
C
      DATA TOL/0.15/,VFMIN,VFMAX/70,300/,TAU,RUNDO/0.4,1.39/
      DATA INTER/5/,TRESH/30/

      SAVE IV,LV,INV,LNV,WAVS,NO,VOISE,TRANS,AMAX

C
C     On doit initialiser le sous programme PPROC qui calcule les
C     periodes du fichier audio.
C

C     Calcul de la valeur maximum du tampon

        DO I=1,NOUT
         IF(ABS(IBUFFO(I)).GT.AMAX) AMAX=ABS(IBUFFO(I))
        END DO

         CLOCK = INTER*1.E-03*SFREQ
         PMIN = SFREQ /VFMAX
         PMAX = SFREQ /VFMIN
         LEVEL = AMAX / TRESH
         DLVEL = AMAX / 10
         TRIG = LEVEL / 3
         SIL = 3
C
C        On remplit les tableaux des donnees avec le sous-programme
C        d'estimation de periodes PPROC.  Par la suite,on lira une
C        ligne a la fois dans les tableaux.

C
C           Filtrage du tampon
C
            CALL FILTRE(IBUFFO,IBUFF,NOUT)

C
C           Traitement du tampon par le sous-programme de detection
C           de periodes.
C

            CALL PPROC(IBUFF,NOUT,PNTLUS,NS,P,C,NP,CI,CO,
     *                 Iw,Lw,Nw)
            NL = Nw
            L = 1
            FINI=.FALSE.

C           Rangement des sorties de PPROC dans l'ordre defini par Iw

         CALL RANGE(NL,Iw,Lw,CI,CO)

C
C     BOUCLE PRINCIPALE
C
      DO WHILE (.NOT.FIN1)

C        Lecture des periodes :
```

```
          CIRC = CI(L)
          ICO = CO(L)
          IwAVE = Iw(L)
          LwAVE = Lw(L)

          L = L + 1

        IF(L.GT.NL) FINI=.TRUE.

C
C       C'est ici que la periode calculee est traitee par rapport a son
C       contexte.  Il y a trois etats possibles: l'etat VOISE lorsque
C       les periodes precedentes se suivent par un ecart d'au plus 15%.
C       L'etat TRANS (transition) lorsqu'une periode ayant un ecart
C       superieur a 15% a ete detectee et l'etat non-voise initialise
C       lorsque que le numero du detecteur de periode (dans PPROC) est 0.

        IF(CIRC.EQ.0) THEN

C           Periode non-voise

           IF(VOISE.OR.TRANS) THEN

            IF(IwAVE.GE.LV)THEN

C             Initialisation d'une periode non-voise

              LNV=LwAVE
              INV=IwAVE
              IF(VOISE) THEN
                 VOISE = .FALSE.

               IF(INIT) THEN
                N=N+1
                LO(N)=LV
                IFO(N)=0
                IPR(N)=1
               ELSE
                 N=N+1
                 LO(N)=LV
                 IFO(N)=SFREQ/(LV-IV)
                 IPR(N)=0
               END IF
                 IF(LV.LT.INV) THEN
                    N=N+1
                    LO(N)=INV
                    IFO(N)=0
                    IPR(N)=1
                 END IF
               ELSE


C                Si la periode non-voise a lieu dans l'etat TRANS, on
C                doit chercher des periodes possibles(ayant un ecart
C                d'au plus 15%) depuis la derniere periode voisee.Si la
C                periode n'a pu etre consideree comme voisee ,elle sera
C                definie comme une transition.


                 TRANS=.FALSE.
```



```
                 NO=NO+1
                 wAVS(NO) = INV

                 LV1=LV
                  DO I=1,NO-1
                    PER=wAVS(I)-LV1
                    IF(ABS(PER-LPER).LE.TOL*LPER) THEN
                      IF(LV.LT.LV1)THEN
                       N=N+1
                       LO(N)=LV1
                       IFO(N)=0
                       IPR(N)=1
                       LV=LV1
                      END IF

                       N=N+1
                       LO(N)=wAVS(I)
                       IFO(N)=SFREQ/(wAVS(I)-LV)
                       IPR(N)=0
                       LV=wAVS(I)
                       END IF
                      LV1=wAVS(I)
                    END DO

                  IF(LV.LT.INV) THEN
                     N=N+1
                     LO(N)=INV
                     IFO(N)=0
                     IPR(N)=1
                     END IF
                  END IF
               END IF
            ELSE

C              Continuation d'une periode non-voisee

               LNV=LwAVE
               END IF
         ELSE

C           La periode actuelle est voisee. Elle sera classee comme
C           periode voisee ou periode de transition suivant le contexte.

            IF(IwAVE.GE.IV) THEN

C              La periode doit etre plus recente que la periode precedente.

               INC=1
               IF(CIRC.LE.3) INC= -1

C              Determination des debuts et fin de la periode actuelle selon
C              les passages par zero.

               ITP=IwAVE
               FTP=LwAVE
               ITMP=FINDZ(ITP,INC,LwAVE)
               FTMP=FINDZ(FTP,INC,IwAVE)

               IF(ITMP.NE.0.AND.FTMP.NE.0) THEN
```

```
C     On continue si la periode est valide. Si les debuts ou
C     fins sont nuls cela indique une erreur dans la recherche
C     des passages par zero.

      PER = FTMP - ITMP
      IF (TRANS) THEN

C        Lorsque l'on est dans l'etat TRANS, on reviendra a
C        l'etat voise seulement si la periode actuelle s'ecarte de
C        la derniere periode a l'interieur d'une limite permise .

         IF(ABS(PER-LPER).LE.TOL*LPER) THEN
            NO=NO+1
            WAVS(NO)=ITMP

C           Lorsque l'on revient a l'etat VOISE on cherche les
C           periodes "possibles" a l'interieur de la zone de
C           transition.On acceptera un ecart de 30%,de plus pour
C           des periodes se trouvant entre deux periodes voisees
C           on acceptera un ecart de 50%.

            DO I=2,NO
              IF(WAVS(I-1).GE.LV.AND.WAVS(I).LE.ITMP) THEN
                PER=WAVS(I)-WAVS(I-1)
                IF(PER.GT.0.7*LPER) THEN
                 IF((WAVS(I-1)-LV).GE.(LPER-TOL*LPER)) THEN
                  IF((WAVS(I-1)-LV).LE.1.5*LPER.OR.
                      (WAVS(I-1)-LV).LE.1.5*PER) THEN
                   N=N+1
                   LO(N)=WAVS(I-1)
                   IFO(N)=SFREQ/(WAVS(I-1)-LV)
                   IPR(N)=0
                  ELSE
                   N=N+1
                   LO(N)=WAVS(I-1)
                   IFO(N)=0
                   IPR(N)=1
                  END IF
                  LV=WAVS(I-1)
                END IF
                 N=N+1
                 LO(N)=WAVS(I)
                 IFO(N)=SFREQ/(WAVS(I)-LV)
                 IPR(N)=0
                 LV=WAVS(I)
              END IF
             END IF
            END DO

           LPER=FTMP-ITMP

           IF(LV.LT.ITMP) THEN
            IF((LPER-(ITMP-LV)).LE.0.5*LPER) THEN
              N=N+1
              LO(N)=ITMP
              IFO(N)=SFREQ/(ITMP-LV)
              IPR(N)=0
            ELSE
              ITMP=LV
            END IF
           END IF
```



```
            TRANS=.FALSE.
            VOISE=.TRUE.
            IV=ITMP
            LV=FTMP
         ELSE

C           Si on reste dans l'etat TRANS, on entre la periode
C           actuelle dans un tableau et si le tableau est
C           rempli, on cherche les periodes "possibles" tout
C           en demeurant dans l'etat de transition.

            IF(NO.GE.MDIM) THEN
               NO=NO+1
               WAVS(NO)=ITMP
               DO I=1,NO
                  PER=WAVS(I)-LV
                  IF(ABS(PER-LPER).LE.TOL*LPER)THEN
                     N=N+1
                     LO(N)=WAVS(I)
                     IFO(N)=0
                     IPR(N)=1
                     LV=WAVS(I)
                     LPER=PER
                     END IF
                  END DO
               NO = 2
               WAVS(1) = ITMP
               WAVS(2) = FTMP
            ELSE
                NO=NO+1
                WAVS(NO)=ITMP

               NO=NO+1
               WAVS(NO) = FTMP
               END IF

            LPER=PER

            END IF
      ELSE IF (VOISE) THEN

C        Lorsque que L'on est dans l'etat voise, on
C        determine si l'ecart entre la periode actuelle
C        et la periode precedente est acceptable et
C        on teste aussi pour les "trous" entre deux
C        periodes consecutives.

         IF(ITMP.GE.IV.AND.FTMP.LT.LV) THEN
          LV=FTMP
          LPER=LV-IV
         END IF

         IF(ITMP.GE.LV) THEN

C           On ne traite pas la meme periode deux fois.

          IF(INIT) THEN
           IF(ABS(PER-(LV-IV)).LE.TOL*PER) THEN
```

```
          LPER=LV-IV
          N=N+1
          LO(N)=LV
          IFO(N)=SFREQ/(LV-IV)
          IPR(N)=0
         ELSE
          N=N+1
          LO(N)=LV
          IFO(N)=0
          IPR(N)=1
          INIT=.FALSE.
         END IF
         IF(LV.LT.ITMP) THEN
          N=N+1
          LO(N)=ITMP
          IFO(N)=0
          IPR(N)=1
         END IF

       ELSE
         IF(LV.LT.ITMP) THEN
          PER=ITMP-LV
          IF(ABS(PER-LPER).LE.2*TOL*LPER.OR.
          ABS(PER-(FTMP-ITMP)).LE.2*TOL*(FTMP-ITMP)) THEN
           N=N+1
           LO(N)=LV
           IFO(N)=SFREQ/(LV-IV)
           IPR(N)=0

           N=N+1
           LO(N)=ITMP
           IFO(N)=SFREQ/(ITMP-LV)
           IPR(N)=0

          ELSE

           IF((LPER-PER).GE.2*TOL*LPER) THEN

            N=N+1
            LO(N)=ITMP
            IFO(N)=SFREQ/(ITMP-IV)
            IPR(N)=0

           ELSE

            N=N+1
            LO(N)=LV
            IFO(N)=SFREQ/(LV-IV)
            IPR(N)=0

            N=N+1
            LO(N)=ITMP
            IFO(N)=0
            IPR(N)=1

           END IF

          END IF

       ELSE
```



```
             N=N+1
             LO(N)=LV
             IFO(N)=SFREQ/(LV-IV)
             IPR(N)=0

            END IF

           END IF

          LV=ITMP
          PER=FTMP-ITMP
C
             Test d'ecart entre les periodes

             IF(ABS(PER-LPER).LE.TOL*LPER.OR.
                INIT)     THEN
C
C               Si on est au debut d'une zone voisee, on peut
C               tolerer de plus grands ecarts tant que les
                detecteurs ne sont pas "accroches".

                IF(INIT.AND.ABS(PER-LPER).LE.TOL*LPER)THEN
                   INIT=.FALSE.
                 END IF
                IV = ITMP
                LV = FTMP
                LPER = LV -IV
              ELSE
C
C               Si la periode actuelle n'a pu etre classee
                comme voisee, on initialise l'etat de transition.

                VOISE = .FALSE.
                TRANS = .TRUE.
                NO = 2
                *AVS(NO-1) = ITMP
                *AVS(NO) = FTMP
                END IF
              END IF
          ELSE
C
C           Ici on teste pour voir si la periode non-voisee
            est terminee.

          IF(ITMP.GT.LO(N).AND.(INV.EQ.0.OR.LWAVE.GT.LNV))THEN
C
              Si oui, on revient a l'etat voise.

              VOISE=.TRUE.
              INIT = .TRUE.
              IV = ITMP
              LV = FTMP
               N=N+1
               LO(N)=ITMP
               IFO(N)=0
               IPR(N)=2
              END IF
            END IF
          END IF
        END IF
      END IF
```

```
        END DO

        IF(.NOT.VOISE.AND..NOT.TRANS.AND.N.EQ.1) THEN
         N=N+1
         LO(N)=LNV
         IFO(N)=0
         IPR(N)=2
        END IF

800 RETURN

        ENTRY INSEG

        INIT=.TRUE.
        TRANS=.FALSE.
        VOISE=.FALSE.
        LNV=0
        AMAX=32767/5

        RETURN

      END

C
C
C
C     SOUS-ROUTINE DE DETERMINATION DES DEBUTS ET FINS DE PERIODES
C
C     La fonction FINDZ retourne la position du point le plus proche
C     de zero a partir de DEB dans la direction indiquee par INC.
C     Cette position doit se situer a un passage par zero. Si la
C     fonction ne peut trouver de passage par zero a l'interieur
C     de l'intervalle specifie par LIM ou si une des extremites
C     du fichier est rencontree, la fonction retourne zero.
C
      INTEGER FUNCTION FINDZ(DEB,INC,LIM)
      IMPLICIT INTEGER (A-Z)
C
      PARAMETER HDR=256,MXLUN=120
      LOGICAL FOUND
      COMMON/FILE_OPEN_SECT/IFAB,MAPADR(2,0:MXLUN-1)
C
      SAVE LDEB,ZDEB,IOFFS,NPNTS
C
      IF(INC.EQ.0) THEN
         FINDZ = 0
         RETURN
         END IF
       IF (LDEB.NE.DEB) THEN
         LSIGN= -INC
         LPOINT=PNT(%VAL((DEB-1)*2+IOFFS))
         FOUND = .FALSE.
         LDEB = DEB
         LH = DEB - ABS(LIM-DEB)
         HG = DEB + ABS(LIM-DEB)
         IF(LH.LE.0) LH = 1
         IF(HG .GT. NPNTS) HG = NPNTS
C
         DO WHILE (.NOT.FOUND)
            DEB = DEB + INC
            IF(DEB.LE.LH.OR.DEB.GE.HG) THEN
               FOUND = .TRUE.
```



```
               ZDEB = 0
            ELSE
               POINT = PNT(%VAL((DEB-1)*2+IOFFS))
               NSIGN = SIGN(1,POINT)
               IF(NSIGN.NE.LSIGN) THEN
                  FOUND = .TRUE.
                  DIF = POINT*NSIGN-LPOINT*LSIGN
                  ZDEB=DEB-INC*(SIGN(1,DIF)+1)*0.5
               ELSE
                  LSIGN = NSIGN
                  LPOINT = POINT
                  END IF
               END IF
            END DO
         END IF
C
      FINDZ = ZDEB
      RETURN
C
C     Entree d'initialisation de FINDZ
C
      ENTRY IFZ(LUN)
C
      LDEB = 0
      ZDEB = 0
      NPNTS = (MAPADR(2,LUN)+2-MAPADR(1,LUN)-2*HDR)/2
      IOFFS = MAPADR(1,LUN) + 2 * HDR
      RETURN

      END
C
C
C
      INTEGER FUNCTION PNT(PAUD)
C
      INTEGER*2 PAUD
C
      PNT = PAUD
C
      RETURN
      END
```

```
C---------------bell-Northern Research / INRS Computer Laboratory-------------
C
C MODULE:       PPROC(IBUFF,BUFL,PNTLUS,NS,PERIOD,ICO,NP,CIRC,ICOW,IWAVE,LWAVE,N
C               INPROC(FREQ,ITFR,INIT)
C
C FILE NAME:    PPROC.FOR
C
C PURPOSE:      EXTRAIRE LES PERIODES D'UN TAMPON AUDIO.
C
C DESCRIPTION:  CE SOUS-PROGRAMME UTILISE LA METHODE DES DETECTEURS
C               PARALLELES DANS LE DOMAINE DU TEMPS POUR DETERMINER
C               LES PERIODES. (POUR DE PLUS AMPLES RENSEIGNEMENTS,
C               ON PEUT CONSULTER L'ARTICLE: Parallel Processing
C               Techniques For Estimating Pitch Periods Of Speech
C               In The Time Domain.  J. ACOUST. SOC. AMER. VOL 46
C               pp442-448 AUG 1969.)
C
C PARAMETERS:
C             SUBROUTINE PPROC:
C
C               IBUFF : TAMPON CONTENANT DES ECHANTILLONS DE PAROLE
C                       PREALABLEMENT FILTRES.
C
C               BUFL : LONGEUR DU TAMPON IBUFF
C
C               PNTLUS : NUMERO DU PREMIER ECHANTILLON DU TAMPON IBUFF
C                        DANS LE FICHIER AUDIO
C
C               NS : LONGUEUR DE LA SECTION DE CHEVAUCHEMENT
C
C            LES PARAMETRES PERIOD,ICO,CIRC,ICOW,IWAVE ET LWAVE
C            SONT DES VECTEURS DE LONGEUR (BUFL/CLOCK+1) OU CLOCK
C            EST DEFINI PLUS LOIN.
C
C       (*)   PERIOD : VECTEUR CONTENANT LES PERIODES ESTIMEES.
C                      LES PERIODES SONT EN UNITES D'ECHANTILLONS.
C
C       (*)   ICO : NOMBRE DE COINCIDENCES OBTENUES POUR CHAQUE
C                   PERIODE ESTIMEE. ICO VARIE DE 0 A 35.
C
C       (*)   CIRC : NO DU DETECTEUR CHOISI POUR CHAQUE PERIODE.
C                    CIRC VARIE DE 0 A 6.  LORSQUE CIRC VAUT 0,
C                    CELA SIGNIFIE UNE PERIODE NON-VOISEE.
C
C       (*)   ICOW : EQUIVALENT A ICO MAIS NE CONTIENT PAS LES
C                    PERIODES NON-VOISEES.
C
C       (*)   IWAVE : DEBUT DE CHAQUE PERIODE VOISEE.
C
C       (*)   LWAVE : FIN DE CHAQUE PERIODE VOISEE.
C
C       (*)   NW : NOMBRE DE VALEURS ENTREES DANS LES VECTEURS:
C                  CIRC,ICOW,IWAVE,LWAVE
C
C           PARAMETRE PASSES EN COMMON:
C
C             TRIG : NIVEAU DE DECLENCHEMENT DES DETECTEURS
C
C             PMIN,PMAX : PERIODES MINIMUM ET MAXIMUM SPECIFIANT
C                         L'INTERVALLE DE PERIODES PERMISES.
```



```
C                         JPMIN,PMAX(
C
C               RUNDO : CONSTANTE DE TEMPS DE L'EXPONENTIELLE UTILISEE
C                       DANS LES DETECTEURS (0.695)
C
C               DLVEL : NIVEAU INITIAL D'UN DETECTEUR APRES UNE PERIODE
C                       NON-VOISEE.
C
C               CLOCK : INTERVALLE DETERMINANT LE TAUX DE CALCUL DES
C                       PERIODES: LES PERIODES SONT ESTIMEES A TOUS
C                       LES 'CLOCK' IEME ECHANTILLONS.
C
C               SIL : NOMBRE DE DETECTEURS REQUIS POUR DETERMINER
C                     UNE PERIODE NON-VOISEE.
C
C               TAU : TAUX DE TEMPS MORT DES DETECTEURS (BLANKING TIME)
C                       (NORMALEMENT 0.4)
C
C               LEVEL : Niveau d'amplitude qui separe le voise du non-voise.
C
C ROUTINES REQUIRED:     AUCUNE
C
C AUTHOR / MAINTAINED BY:   R. MARCHAND
C
C DATE CREATED:   10 AOUT 1981
C
C UPDATES:      MODIFICATION DES ARGUMENTS D'ENTREE DU SOUS PROGRAMME
C
C               MODIFICATION DU CHOIX DES DEBUT ET FIN D'UNE PERIODE
C               VOISEE:CEUX CI SONT CHOISIS COMME ETANT LES PLUS PROCHES
C               DE LA FIN DE LA PERIODE PRECEDENTE.
C
C               INTRODUCTION D'UNE CONTRAINTE SUR LA LONGUEUR D'UNE
C               SECTION NON VOISEE : CELLE CI DOIT ETRE STRICTEMENT
C               SUPERIEURE A CLOCK.
C
C                   C. SIDE - 10 DECEMBRE 1981
C
C---------------bell-Northern Research / INRS Computer Laboratory-------------
C
      SUBROUTINE PPROC(IBUFF,BUFL,PNTLUS,NS,PERIOD,ICO,NP,CIRC,ICOW,
     *                 IWAVE,LWAVE,NW)
C
      IMPLICIT INTEGER (A-Z)
C
      COMMON/PARAM/TRIG,PMIN,PMAX,RUNDO,DLVEL,CLOCK,SIL,TAU,LEVEL
      REAL RUNDO,TAU
C
      REAL FREQ,VRANGE,VCO,BETA
      DIMENSION VRANGE(2,4),RANGE(2,4),CO(4,4),VCO(4,4)
      DIMENSION PP(6,6),PPE(36),BIAS(4),BLANK(6),LSTIME(6),
     *          PEAK(6),BETA(6),ME(6),ITV(6),FTV(6)
      DIMENSION IBUFF(BUFL),PERIOD(BUFL/CLOCK+1),
     *          ICO(BUFL/CLOCK+1),IWAVE(BUFL/CLOCK+1),
     *          LWAVE(BUFL/CLOCK+1),CIRC(BUFL/CLOCK+1),
     *          ICOW(BUFL/CLOCK+1)
      LOGICAL DET(6)
C
      EQUIVALENCE (PP(1,1),PPE(1))
C
```

```
      DATA VRANGE/1.6,3.1,3.1,6.3,6.3,12.7,12.7,25.5/
      DATA VCO/0.1,0.2,0.3,0.4,0.2,0.4,0.6,0.8,0.4,
     *         0.8,1.2,1.6,0.8,1.6,2.4,3.2/
      DATA BIAS/1,2,5,7/
C
      SAVE LSTPNT,NEWCRP,NEWCRN,BLANK,LSTIME,
     *     PP,PEAK,BETA,RANGE,CO,LSTP,ITNV,ITV,FTV,MAXP
C     Fonction exponentielle decroissante
      DECAY(NO,TIME)=PEAK(NO)*EXP(-BETA(NO)*(TIME-BLANK(NO)))
C
C
      IT = 1
      ITT = 1

        TEMPS=PNTLUS
        IDEB=1
        IF(TEMPS.LT.0) THEN
         TEMPS=0
         IDEB=IDEB-PNTLUS
        END IF

      DO J=IDEB,BUFL=NS
         TEMPS = TEMPS + 1
         IF(ABS(IBUFF(J)).GT.MAXP)MAXP=ABS(IBUFF(J))
         IF (IBUFF(J).GE.0) THEN
C
C           Boucle de recherche de cretes positives
C
            IF(IBUFF(J).LT.LSTPNT) THEN
               Une crete positive a ete detectee.
               LSTCRP=NEWCRP
               NEWCRP=LSTPNT
               ME(1)=NEWCRP
               IF(ME(1).LE.0) ME(1)=0
               ME(2)=NEWCRP-NEWCRN
               ME(3)=NEWCRP-LSTCRP
               IF(ME(3).LE.0) ME(3)=0
               On regarde si la crete declenche des detecteurs.
               DO N=1,3
                DET(N)=.FALSE.
                  IF(ME(N).GT.TRIG.AND.TEMPS.GT.
     *               BLANK(N).AND.ME(N).GT.
     *               DECAY(N,TEMPS))          THEN
                     Si oui, le detecteur no N est reinitialise.
                     DET(N)=.TRUE.
                     ITV(N) = LSTIME(N)
                     FTV(N) = TEMPS
                     Periode actuelle
                     PNEW= TEMPS-LSTIME(N)
                     LSTIME(N)=TEMPS
                     Nouvelle periode
                     PER=(PP(N,1)+PNEW)/2
                     PP(N,3)=PP(N,2)
                     PP(N,2)=PP(N,1)
                     PP(N,1)=PER
                     La periode doit demeurer dans les limites permises.
                     IF(PER.LT.PMIN)PP(N,1)=PMIN
                     IF(PER.GT.PMAX)PP(N,1)=PMAX
                     Nouvelles caracteristiques du detecteur.
                     PEAK(N)=ME(N)
                     BLANK(N)=TEMPS+TAU*PP(N,I)
```



```
                     BETA(N)=RUNDO/PP(N,1)
                     END IF
                  END DO
               END IF
         ELSE
C
C           Boucle de recherche des cretes negatives.
C
            IF(IBUFF(J).GT.LSTPNT) THEN
               Une crete negative a ete detectee.
               LSTCRN = NEWCRN
               NEWCRN = LSTPNT
               ME(4) = -NEWCRN
               IF(ME(4).LE.0) ME(4)=0
               ME(5)=NEWCRP-NEWCRN
               ME(6)=LSTCRN-NEWCRN
               IF(ME(6).LE.0) ME(6)=0
C
C              Ici, c'est le meme cheminement que pour une crete
C              positive sauf que ce sont les detecteurs negatifs
C              qui sont verifies.
C
               DO N=4,6
                DET(N)=.FALSE.
                  IF(ME(N).GT.TRIG.AND.TEMPS
     *                .GT.BLANK(N).AND.ME(N).GT.
     *                DECAY(N,TEMPS))       THEN
                     DET(N)=.TRUE.
                     ITV(N)=LSTIME(N)
                     FTV(N)=TEMPS
                     PNEW=TEMPS-LSTIME(N)
                     LSTIME(N)=TEMPS
                     PER=(PP(N,1)+PNEW)/2
                     PP(N,3)=PP(N,2)
                     PP(N,2)=PP(N,1)
                     PP(N,1)=PER
                     IF(PER.LT.PMIN)PP(N,1)=PMIN
                     IF(PER.GT.PMAX)PP(N,1)=PMAX
                     PEAK(N)=ME(N)
                     BLANK(N)=TEMPS+TAU*PP(N,1)
                     BETA(N)=RUNDO/PP(N,1)
                     END IF
                  END DO
               END IF
            END IF
C        Test de periode non-voisee pour tous les detecteurs.
         DO N=1,6
            IF(BLANK(N)+PMAX.LE.TEMPS) THEN
C              Le detecteur N est reinitialise avec la periode maximum.
               PNEW=TEMPS-BLANK(N)
               PEAK(N)=DLVEL
               LSTIME(N)=TEMPS
               BETA(N)=RUNDO/PMAX
               BLANK(N)=TEMPS+TAU*PMAX
               PP(N,3)=PP(N,2)
               PP(N,2)=PP(N,1)
               PP(N,1)=PMAX
               END IF
            END DO
         IF(JMOD(TEMPS,CLOCK).LT.1) THEN
C           Il est temps de calculer la periode
```

```
      TEST = 0
C     Etablissement du tableau qui servira a determiner la periode.
      DO N=1,6
         PP(N,4)=PP(N,1)+PP(N,2)
         PP(N,5)=PP(N,2)+PP(N,3)
         PP(N,6)=PP(N,1)+PP(N,2)+PP(N,3)
         END DO
      GMAX=0
      COUNT=0
      MAXCO=0
      DO K=1,6
         ELEM=PP(K,1)
C        Test de periode non-voisee
         IF(ELEM.EQ.PMAX) TEST=TEST+1
C        Determination de la table de tolerance.
         DO R=1,4
            IF(ELEM.GE.RANGE(1,R).AND.ELEM.LT.
     *         RANGE(2,R)) TAB=R
            END DO
         DO L=1,4
            TOL = CO(L,TAB)
C           Calcul des coincidences
            DO M=1,36
               IF(ABS(ELEM-PPE(M)).LE.TOL)COUNT=COUNT+1
               END DO
C           Correction en regard du niveau de tolerance.
            COUNT=COUNT-BIAS(L)-1
C           Determination du maximum de coincidences pour
C           un element particulier.
            IF(COUNT.GT.MAXCO) MAXCO=COUNT
            COUNT=0
            END DO
C        Determination du maximum de coincidences absolu.
         IF(MAXCO.GT.GMAX) THEN
C           Enregistrement du gagnant
            GMAX=MAXCO
            PERIOD(IT)=ELEM
            ICO(IT) = GMAX
            IF(.NOT.DET(K).AND.SK.EQ.K) THEN
            DO KP=1,6
             IF(SK.NE.KP.AND.ABS(ELEM-PP(KP,1)).LE.TOL) SK=KP
            END DO
            ELSE
             SK=K
            END IF
            IF(ITT.GT.1.AND.(ITV(SK)-LWAVE(ITT-1)).GT.0.15*ELEM) THEN
            DO KP =1,6
             IF(ABS(ELEM-PP(KP,1)).LE.TOL.AND.ITV(KP).LT.ITV(SK)) SK=KP
            END DO
            END IF
         END IF
      MAXCO = 0
      END DO
C     Periode non-voisee
      IF(TEST.GE.SIL.OR.MAXP.LT.LEVEL) PERIOD(IT) = 0
C     Calcul de l'intervalle de temps non-voise.
      IF(PERIOD(IT).EQ.0.AND.LSTP.NE.0.OR.
     *   PERIOD(IT).NE.0.AND.LSTP.EQ.0) THEN
         LSTP = PERIOD(IT)
         IF(PERIOD(IT).EQ.0) THEN
            ITNV = TEMPS
         ELSE
          IF((TEMPS-ITNV).GT.CLOCK) THEN
C           Enregistrememt d'un intervalle non-voise.
            FINV = TEMPS
            LWAVE(ITT)=ITNV
            LWAVE(ITT)=FINV
            CIRC(ITT)=0
            ICOW(ITT)=0
            ITT=ITT+1
           END IF
          END IF
         END IF
        IF(PERIOD(IT).NE.0) THEN
C           Enregistrement des debut et fin de periode avec le
C           detecteur utilise et le nombre de coincidences.
           LWAVE(ITT)=ITV(SK)
           LWAVE(ITT)=FTV(SK)
           CIRC(ITT)=SK
           ICOW(ITT)=ICO(IT)
           ITT=ITT+1
           END IF
        IT = IT + 1
        MAXP = 0
        END IF
       LSTPNT= IBUFF(J)
       END DO

      IF(LSTP.EQ.0) THEN
       LWAVE(ITT)=ITNV
       LWAVE(ITT)=TEMPS
       CIRC(ITT)=0
       ICOW(ITT)=0
       ITNV=LWAVE(ITT)
       ITT=ITT+1
      END IF

C
      NW = ITT - 1
      NP = IT - 1
      RETURN
C
C     Sous-routine d'initialisation
C
      ENTRY INPROC (FREQ,ITER,INIT)
C
      MAXP = 0
      LSTPNT=0
      NEWCRP=0
      NEWCRW=0
      LSTP=0
C
      DO N=1,6
         BLANK(N)=0
         LSTIME(N)=0
         PEAK(N)=0
         BETA(N)=0
         DO J=1,6
            PP(N,J)=0
            END DO
         END DO
C
```

```
C     Conversion de differentes constantes en unites d'echantillons.
C
      IF(INIT  .NE. 0) THEN
         DO R=1,4
            RANGE(1,R)=VRANGE(1,R)*1.E-03*FREQ
            RANGE(2,R)=VRANGE(2,R)*1.E-03*FREQ
            END DO
         DO L=1,4
            DO TAB=1,4
               CO(L,TAB)=VCO(L,TAB)*1.E-03*FREQ
               END DO
            END DO
         END IF
      RETURN
      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       ACORR (X, NPTS, RXX, NTERM)
*
*
* PURPOSE:
*       THIS SUBROUTINE CALCULATES THE AUTO-CORRELATION FOR
*       A DATA VECTOR.
*
*
* DESCRIPTION:
*       THIS SUBROUTINE CALCULATES THE AUTO-CORRELATION FOR A
*       GIVEN DATA VECTOR. RXX(I) GIVES THE AUTO-CORRELATION AT
*       LAG I-1 FOR I RUNNING FROM 1 TO NTERM.
*
*                  NPTS
*         RXX(I) = SUM X(J) * X(J-I+1)
*                  J=I
*
*
* PARAMETERS:
*       X       - INPUT DATA VECTOR WITH NPTS ELEMENTS
*       NPTS    - NUMBER OF DATA POINTS
*   (*) RXX     - AUTO-CORRELATION VECTOR WITH NTERM ELEMENTS
*       NTERM   - NUMBER OF AUTO-CORRELATION TERMS
*
*
* ROUTINES REQUIRED:
*       NONE
*
*
* AUTHOR / MAINTAINED BY:
*       P. KABAL
*
*
* DATE CREATED:
*       81/02/25
*
*
* UPDATES:
*
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
```

```
      SUBROUTINE ACORR(X,NPTS,RXX,NTERM)

      REAL X(NPTS),RXX(NTERM)

      DO 160 I=1,NTERM
        SUM=0.0
        IF(I.GT.NPTS) GO TO 140

        DO 120 J=1,NPTS
          SUM=SUM+X(J)*X(J-I+1)
  120   CONTINUE

        RXX(I)=SUM
  140 CONTINUE

      RETURN

      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       SUBROUTINE AUTOK (RXX, NPOLE, REFL)
*
*
* PURPOSE:
*       This routine solves a set of toeplitz equations to obtain
*       a set of reflection coefficients.
*
*
* DESCRIPTION:
*       This subroutine implements a modified form of Durbin's
*       algorithm for solving a set of auto-correlation equations.
*       This routine uses an intermediate variable which is
*       constrained to be less than the energy of the signal
*       i.e. RXX(1). This variable takes the role of the predictor
*       coefficients which normally are used in durbin's form
*       of the algorithm. This routine returns only the
*       reflection coefficients.
*         Reference: J. Leroux and C.J. Gueguen, "A Fixed Point
*                    Computation of the Partial Correlation
*                    Coefficients", IEEE Trans. ASSP, June 1977,
*                    pp. 257-259.
*
*
* PARAMETERS:
*       RXX     - Array of NPOLE+1 auto-correlation coefficients
*       NPOLE   - Number of reflection coefficients (maximum 13)
*    (*) REFL   - Output array of NPOLE reflection coefficients
*
*
* ROUTINES REQUIRED:
*       None
*
*
* AUTHOR / MAINTAINED BY:
*       P. Kabal
*
*
* DATE CREATED:
*       20-08-80
*
*
* UPDATES:
*       81/02/24  FORTRAN 77 version
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
```

```
      SUBROUTINE AUTOK (RXX, NPOLE, REFL)

      PARAMETER (MXPOLE=13)

      REAL RXX(NPOLE+1),REFL(NPOLE)
      REAL EP(MXPOLE),EN(MXPOLE)

* Initialization
      DO 140 I=1,NPOLE
        EP(I)=RXX(I+1)
        EN(I)=RXX(I)
140   CONTINUE
      IF (EN(1).LE.0.0) EN(1)=1.0

* Recursive solution of the equations
      DO 180 K=1,NPOLE
        REFL(K)=-EP(K)/EN(1)

        IF (K.EQ.NPOLE) GO TO 900

        EN(1)=EN(1)+REFL(K)*EP(K)
        EP(NPOLE)=EP(NPOLE)+REFL(K)*EN(NPOLE-K+1)

        L=2
        DO 160 I=K+1,NPOLE-1
          SUM=EP(I)+REFL(K)*EN(L)
          EN(L)=EN(L)+REFL(K)*EP(I)
          EP(I)=SUM
          L=L+1
160     CONTINUE

180   CONTINUE

900   RETURN

      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       DEEM (X, Y, NPTS, PRE, AMEM)
*
*
* PURPOSE:
*       THIS SUBROUTINE DE-EMPHASIZES A DATA VECTOR.
*
*
* DESCRIPTION:
*       THIS SUBROUTINE IMPLEMENTS A FIRST ORDER RECURSIVE FILTER
*       TO DE-EMPHASIZE A SIGNAL VECTOR.
*
*         Y(I) = X(I) + PRE * Y(I-1))
*
* PARAMETERS:
*       X       - INPUT VECTOR WITH NPTS ELEMENTS
*    (*) Y      - OUTPUT VECTOR WITH NPTS ELEMENTS. THE VECTOR Y MAY
*                 BE THE SAME VECTOR AS X.
*       NPTS    - NUMBER OF POINTS IN EACH OF X AND Y.
*       PRE     - PRE-EMPHASIS FACTOR (NORMALLY POSITIVE).
*    (*) AMEM   - MEMORY ELEMENT. THIS LOCATION IS USED TO STORE THE
*                 THE LAST OUTPUT ELEMENT. INITIALLY, AMEM SHOULD BE
*                 SET TO ZERO. AS A LONG ARRAY OF DATA IS PROCESSED
*                 BLOCK BY BLOCK, THE VALUE OF AMEM RETURNED FROM ONE
*                 CALL TO THIS SUBROUTINE IS SUITABLE AS THE INITIAL
*                 VALUE FOR THE NEXT CALL.
*
* ROUTINES REQUIRED:
*       NONE
*
*
* AUTHOR / MAINTAINED BY:
*       P. KABAL
*
*
* DATE CREATED:
*       81/02/28
*
*
* UPDATES:
*
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
```

```
      SUBROUTINE DEEM (X, Y, NPTS, PRE, AMEM)

      REAL X(NPTS),Y(NPTS)

      DO 100 I=1,NPTS
        AMEM=X(I)+PRE*AMEM
        Y(I)=AMEM
100   CONTINUE

      RETURN

      END
```

```
C--------------Bell-Northern Research / INRS Computer Laboratory------------
C
C
C MODULE:       INFIL(FILCOF,NTAP)
C               FILTRE(IBNF,IBF,BUFL)
C
C FILE NAME:    FILTRE.FOR
C
C
C PURPOSE:      FILTRER UN TAMPON DE FICHIER AUDIO
C
C
C DESCRIPTION:  FILTRE RECOIT UN TAMPON IBNF ET RENVOIT SON IMAGE FILTREE
C               DANS IBF.
C
C PARAMETERS:
C                SOUS-ROUTINE INFIL:
C
C                  FILCOF : NOM DU FICHIER DE COEFFICIENTS FIR
C
C           (*)    NTAP : NOMBRE DE COEFFICIENTS FIR: RETOURNE AU
C                         PROGRAMME APPELANT.
C
C                SOUS-ROUTINE FILTRE:
C
C                  IBNF : TAMPON DE POINTS NON FILTRES.
C
C           (*)    IBF : TAMPON DE POINTS FILTRES.
C
C                  BUFL : NOMBRE DE POINTS A FILTRER.
C
C            LE TAMPON IBNF DOIT ETRE SOUS LA FORME SUIVANTE:
C
C      NTAP/2       BUFL          NTAP/2
C      ***********************************
C      * AVANT *     ACTUEL     * APRES *
C      ***********************************
C
C                     ***********************************
C                     * AVANT *     ACTUEL     * APRES *
C                     ***********************************
C
C              OU "ACTUEL" EST L'IMAGE NON-FILTREE DU TAMPON IBF.
C
C ROUTINES REQUIRED:  DEFNAM
C
C AUTHOR / MAINTAINED BY:   R. MARCHAND
C
C DATE CREATED:     10 AOUT 1981
C
C UPDATES:
C
C
C--------------Bell-Northern Research / INRS Computer Laboratory------------
C
      SUBROUTINE FILTRE(IBNF,IBF,BUFL)
C
      PARAMETER MAXCOF=255
      INTEGER BUFL,FILCOF
      CHARACTER*80 FILCOF,FCOF
      DIMENSION HTAB(MAXCOF),IBNF(BUFL+MAXCOF-1)
      DIMENSION IBF(BUFL)
C
      SAVE NCOF
C
C     Convolution
C
      DO I=1,BUFL
         SUM = 0.0
         DO J=1,NCOF
            SUM=SUM+HTAB(J)*IBNF(I+J-1)
            END DO
         IBF(I)=SUM
         END DO
      RETURN
C
C     Sous-routine d'initialisation
C
      ENTRY INFIL(FILCOF,NTAP)
C
      CALL DEFNAM(FILCOF,FCOF,NCHR,'.COF')
      CALL LIBSGET_LUN(FILCOF)
      IF(FILCOF .LE. 0) GO TO 888
      OPEN(UNIT=FILCOF,NAME=FCOF,TYPE='OLD',READONLY,ERR=888)
C
      I=1
   11 READ(FILCOF,100,END=21) HTAB(I)
      I=I+1
      GO TO 11
   21 NTAP = I - 1
      NCOF=NTAP
      CLOSE (UNIT=FILCOF)
      CALL LIBSFREE_LUN(FILCOF)
      RETURN
  888 NTAP = 0
      RETURN
  100 FORMAT(E16.0)
      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory------------
*
*
* MODULE:
*       ISHIFT (IX, NKEEP, NSHIFT)
*
*
* PURPOSE:
*       THIS ROUTINE SHIFTS THE ELEMENTS OF AN INTEGER VECTOR.
*
*
* DESCRIPTION:
*       THIS ROUTINE SHIFTS THE ELEMENTS OF AN INTEGER VECTOR.
*       IF THE NUMBER OF ELEMENTS TO BE RETAINED IS ZERO, NO
*       ACTION IS TAKEN.
*
*       FOR A SHIFT DOWN (NSHIFT POSITIVE), THE ELEMENTS OF
*       THE ARRAY ARE SHIFTED DOWN TO THE BOTTOM OF THE
*       ARRAY. THE INPUT ARRAY MUST HAVE AT LEAST
*       NKEEP+NSHIFT ELEMENTS.
*         IX(NSHIFT+I) --> IX(I) , FOR I FROM 1 TO NKEEP.
*
*       FOR A SHIFT UP (NSHIFT NEGATIVE), THE ELEMENTS OF
*       THE ARRAY ARE SHIFTED UP TO THE TOP OF THE ARRAY.
*       THE INPUT ARRAY MUST HAVE AT LEAST NKEEP-NSHIFT
*       ELEMENTS (RECALL THAT NSHIFT IS NEGATIVE).
*         IX(I) --> IX(I-NSHIFT), FOR I FROM NKEEP TO 1.
*
*
* PARAMETERS:
*       IX     - INPUT ARRAY
*       NKEEP  - NUMBER OF ELEMENTS TO BE RETAINED
*       NSHIFT - NUMBER OF POSITIONS TO BE SHIFTED. NSHIFT
*                IS POSITIVE FOR A SHIFT DOWN AND NEGATIVE
*                FOR A SHIFT UP. NSHIFT EQUAL TO ZERO IS
*                ALSO PERMISSIBLE.
*
*
* ROUTINES REQUIRED:
*       NONE
*
*
* AUTHOR / MAINTAINED BY:
*       P. KABAL
*
*
* DATE CREATED:
*       81/02/15
*
*
* UPDATES:
*
*
*---------------Bell-Northern Research / INRS Computer Laboratory------------
      SUBROUTINE ISHIFT (IX, NKEEP, NSHIFT)

      INTEGER IX(*)

      IF (NKEEP.LE.0) GO TO 900

* SHIFT DOWN
      IF (NSHIFT.LT.0) GO TO 200
      DO 100 I=1,NKEEP
        IX(I)=IX(NSHIFT+I)
100   CONTINUE
      GO TO 900

* SHIFT UP
200   DO 300 I=NKEEP,1,-1
        IX(I-NSHIFT)=IX(I)
300   CONTINUE

900   RETURN

      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory------------
*
*
* MODULE:
*       LATSYN (FIRST, NPOLES, REFL, X, NPTS, PRE, XOUT)
*
* PURPOSE:
*       SYNTHESIZE A FRAME OF OUTPUT POINTS FOR A GIVEN SET OF
*       REFLECTION COEFFICIENTS.
*
* DESCRIPTION:
*       AN INPUT EXCITATION SIGNAL IS FILTERED USING A LATTICE
*       FILTER WITH A GIVEN SET OF REFLECTION COEFFICIENTS.
*
* PARAMETERS:
*       NPOLES  - NUMBER OF REFLECTION COEFFICIENTS
*       REFL    - ARRAY OF REFLECTION COEFFICIENTS
*       X       - ARRAY CONTAINING THE EXCITATION SIGNAL
*       NPTS    - NO. OF POINTS IN FRAME
*  (*) RCMEM    - ARRAY OF NPOLES ELEMENTS CONTAINING THE FILTER MEMORY.
*                 THIS ARRAY IS MODIFIED ON OUTPUT. INITIALLY IT SHOULD BE
*                 SET TO ZERO.
*  (*) XOUT     - OUTPUT ARRAY
*
* ROUTINES REQUIRED:
*       NONE
*
* AUTHOR / MAINTAINED BY:
*       D.C. STEVENSON
*
* DATE CREATED:
*       3-APR-80
*
* UPDATES:
*       81/11/04  Fix dimension of RCMEM
*
*---------------Bell-Northern Research / INRS Computer Laboratory------------

      SUBROUTINE LATSYN (NPOLES, REFL, X, NPTS, RCMEM, XOUT)

      REAL REFL(NPOLES),X(NPTS),XOUT(NPTS),RCMEM(NPOLES)

      DO 200 I=1,NPTS
        EP=X(I)-REFL(NPOLES)*RCMEM(NPOLES)

        DO 300 K=NPOLES-1,1,-1
          EP=EP-REFL(K)*RCMEM(K)
          RCMEM(K+1)=RCMEM(K)+REFL(K)*EP
300     CONTINUE

        RCMEM(1)=EP
        XOUT(I)=EP
200   CONTINUE

99    RETURN

      END
```

```
C---------------bell-Northern Research / INRS Computer Laboratory-------------
C
C MODULE:            SUBROUTINE LISS(ITAB,IDIM)
C                    INTEGER FUNCTION MED(IVECT,LVECT)
C
C FILE NAME:         LISS.FOR
C
C PURPOSE:       LISSER UN TABLEAU DE NOMBRE ENTIERS
C
C DESCRIPTION:   LE TABLEAU EST LISSE D'ABORD PAR UNE MEDIANE D'ORDRE 3
C                PUIS PAR UNE MEDIANE D'ORDRE 5.
C
C PARAMETERS:
C               SUBROUTINE LISS:
C
C            (*)     ITAB : TABLEAU DE NOMBRE ENTIERS A LISSER.
C                           LES NOMBRES LISSES SONT RETOURNES DANS
C                           LE MEME TABLEAU.
C
C                    IDIM : DIMENSION DU TABLEAU ITNL
C
C                 FONCTION MED:
C
C                    IVECT : VECTEUR DE NOMBRES ENTIERS A METTRE EN ORDRE.
C
C                    LVECT : LONGUEUR DU VECTEUR.
C
C                  LA FONCTION RETOURNE AVEC LA VALEUR MEDIANE DU VECTEUR
C                ORDONNE.
C
C ROUTINES REQUIRED:     AUCUNE
C
C AUTHOR / MAINTAINED BY:    R.   MARCHAND
C
C DATE CREATED:      17 AOUT 1981
C
C UPDATES:
C
C
C---------------bell-Northern Research / INRS Computer Laboratory-------------
C
      SUBROUTINE LISS (ITAB,IDIM)
C
      DIMENSION ITAB(IDIM),IV(5)
      INTEGER MED
C
      IV(2)=ITAB(1)
      IV(3)=ITAB(2)
C     Lissage avec une mediane d'ordre 3.
      DO I=3,IDIM
         DO K=1,2
            IV(K)=IV(K+1)
            END DO
         IV(3)=ITAB(I)
         ITAB(I-1)=MED(IV,3)
         END DO
C
      IV(2)=ITAB(I)
      IV(3)=ITAB(2)
      IV(4)=ITAB(3)
      IV(5)=ITAB(4)
```



```
C     Lissage avec une mediane d'ordre 5.
      DO I=5,IDIM
         DO K=1,4
            IV(K)=IV(K+1)
            END DO
         IV(5)=ITAB(I)
         ITAB(I-2)=MED(IV,5)
         END DO
C
      RETURN
      END
C
C
      INTEGER FUNCTION MED(IVECT,LVECT)
C
      DIMENSION IVECT(LVECT),IT(25)
      INTEGER TEMP
      LOGICAL PERMUT
C
C     Transfert du vecteur IVECT dans un vecteur de travail.
C
      DO I=1,LVECT
         IT(I)=IVECT(I)
         END DO
C
      PERMUT=.TRUE.
C
C     Tri
C
      DO WHILE (PERMUT)
         PERMUT=.FALSE.
         DO I=1,LVECT-1
            IF(IT(I).GT.IT(I+1)) THEN
               TEMP = IT(I)
               IT(I) = IT(I+1)
               IT(I+1) = TEMP
               PERMUT=.TRUE.
               END IF
            END DO
         END DO
C
      MED = IT(LVECT/2+1)
C
      RETURN
      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       PREEM (X, Y, NPTS, PRE, AMEM)
*
*
* PURPOSE:
*       THIS SUBROUTINE PRE-EMPHASIZES A DATA VECTOR.
*
*
* DESCRIPTION:
*       THIS SUBROUTINE FORMS THE FIRST DIFFERENCE WITH THE PAST
*       SAMPLE WEIGHTED BY PRE OF A VECTOR TO FORM A NEW VECTOR.
*
*         Y(I) = X(I) - PRE * X(I-1)
*
* PARAMETERS:
*       X       - INPUT VECTOR WITH NPTS ELEMENTS
*   (*) Y       - OUTPUT VECTOR WITH NPTS ELEMENTS. THE VECTOR Y MAY
*                 BE THE SAME VECTOR AS X.
*       NPTS    - NUMBER OF POINTS IN EACH OF X AND Y.
*       PRE     - PRE-EMPHASIS FACTOR (NORMALLY POSITIVE).
*   (*) AMEM    - MEMORY ELEMENT. THIS LOCATION IS USED TO STORE THE
*                 THE LAST INPUT ELEMENT. INITIALLY, AMEM SHOULD BE
*                 SET TO ZERO. AS A LONG ARRAY OF DATA IS PROCESSED
*                 BLOCK BY BLOCK, THE VALUE OF AMEM RETURNED FROM ONE
*                 CALL TO THIS SUBROUTINE IS SUITABLE AS THE INITIAL
*                 VALUE FOR THE NEXT CALL.
*
* ROUTINES REQUIRED:
*       NONE
*
*
* AUTHOR / MAINTAINED BY:
*       P. KABAL
*
*
* DATE CREATED:
*       81/02/25
*
*
* UPDATES:
*
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
      SUBROUTINE PREEM(X,Y,NPTS,PRE,AMEM)

      REAL X(NPTS),Y(NPTS)

      DO 100 I=1,NPTS
        TEMP=X(I)
        Y(I)=X(I)-PRE*AMEM
        AMEM=TEMP
100   CONTINUE

      RETURN

      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory------------
*
*
* MODULE:
*       SUBROUTINE QNTDFN (XQ, YQ, NLEV, QTYPE, PARM)
*
* PURPOSE:
*       This routine generates quantization tables.
*
* DESCRIPTION:
*       This routine returns the break points and the output levels for
*       a given type of quantizer.
*
* PARAMETERS:
*   (*) XQ      - Output array of NLEV-1 quantizer break points which define
*                 the NLEV quantizer regions. The values are in ascending order.
*   (*) YQ      - Output array of NLEV quantizer output values. The
*                 quantizer output levels are in ascending order.
*       NLEV    - Number of quantizer output regions (maximum 64 for
*                 Gaussian, Laplace or gamma quantizers (QTYPE equal to
*                 'Normal', 'Laplace' OR 'Gamma'))
*       QTYPE   - Character designating the type of quantizer desired. Only
*                 the first character is significant.
*                 'A-law'   - A-law quantizer
*                 'Mu-law'  - Mu-law quantizer
*                 'Uniform' - Uniform quantizer
*                 'Normal'  - Minimum mean square error quantizer for a
*                             zero mean Gaussian (normal) probability
*                             density function
*                 'Laplace' - Minimum mean square error quantizer for a
*                             zero mean Laplacian probability density
*                             function
*                 'Gamma'   - Minimum mean square error quantizer for a
*                             zero mean gamma probability density function
*       PARM    - Input parameter. This parameter has different meanings
*                 for the different values of QTYPE.
*                 'A-law'   - Parameter A of the A-law quantizer (greater
*                             than or equal to 1). The quantizer input is
*                             assumed to lie between -1 and +1.
*                 'Mu-law'  - Parameter mu of the mu-law quantizer (greater
*                             than zero). The quantizer input is assumed to
*                             lie between -1 and +1.
*                 'Uniform' - The maximum value of the quantizer input,
*                             i.e. the input is assumed to lie between
*                             -PARM and +PARM.
*                 'Normal'  - Standard deviation of the Gaussian density
*                             function
*                 'Laplace' - Standard deviation of the Laplacian density
*                             function
*                 'Gamma'   - Standard deviation of the gamma density
*                             function
*
* ROUTINES REQUIRED:
*       ALAW    - A-law companding law
*       LIBSGET_LUN - Get a unit number
*       LIBSFREE_LUN - Release a unit number
*       ULAW    - Mu-law companding law
*
*
* SPECIAL REQUIREMENTS:
*       The logical name QUANT is used to specify the device and directory
*       of the quantizer files.
*
*
* AUTHOR / MAINTAINED BY:
*       P. Kabal
*
*
* DATE CREATED:
*       79/05/24
*
*
* UPDATES:
*       81/12/12  Unit number argument removed
*
*---------------Bell-Northern Research / INRS Computer Laboratory------------
```

```
      SUBROUTINE QNTDFN (XQ, YQ, NLEV, QTYPE, PARM)

      CHARACTER*(*) CALAW,MULAW,UNIF,GAUSS,LAPLC,GAMMA
      PARAMETER (CALAW='A',MULAW='M',UNIF='U',GAUSS='N',LAPLC='L',
     -           GAMMA='G')

      CHARACTER*1 QTYPE
      CHARACTER*9 FNAME

      REAL XQ(NLEV-1),YQ(NLEV)


      IF (QTYPE.EQ.GAUSS .OR. QTYPE.EQ.LAPLC .OR. QTYPE.EQ.GAMMA) THEN

* Minimum mean square error quantizers
        IF (QTYPE.EQ.GAUSS) THEN
          FNAME='GAUSS'
        ELSE IF (QTYPE.EQ.LAPLC) THEN
          FNAME='LAPLACE'
        ELSE
          FNAME='GAMMA'
        END IF

* Read the quantizer levels from the appropriate file
        CALL LIB$GET_LUN(LUN)
        OPEN (UNIT=LUN,ACCESS='DIRECT',STATUS='OLD',READONLY,
     -        FILE='QUANT:'//FNAME//'.QNT')
        READ (UNIT=LUN,REC=NLEV) YQ
        CLOSE (UNIT=LUN)
        CALL LIB$FREE_LUN(LUN)

* Scale the quantizer
        DO 200 I=1,NLEV
          YQ(I)=PARM*YQ(I)
200     CONTINUE

* Generate the break points
        DO 220 I=1,NLEV-1
          XQ(I)=0.5*(YQ(I)+YQ(I+1))
220     CONTINUE


      ELSE

* A-law, mu-law or uniform quantizer
        IF (QTYPE.EQ.UNIF) THEN
          XMAX=PARM
        ELSE
          XMAX=1.0
        END IF

* Generate the output levels
        DO 300 I=1,NLEV
          YQQ=(2*I-(NLEV+1)) *XMAX/NLEV

          IF (QTYPE.EQ.MULAW) THEN
            YQ(I)=ULAW(YQQ,PARM)
          ELSE IF (QTYPE.EQ.CALAW) THEN
            YQ(I)=ALAW(YQQ,PARM)
```



```
          ELSE


            YQ(I)=YQQ
          END IF

300     CONTINUE

* Generate the break points
        DO 320 I=1,NLEV-1
          XQQ=(2*I-NLEV) *XMAX/NLEV

          IF (QTYPE.EQ.MULAW) THEN
            XQ(I)=ULAW(XQQ,PARM)
          ELSE IF (QTYPE.EQ.CALAW) THEN
            XQ(I)=ALAW(XQQ,PARM)
          ELSE
            XQ(I)=XQQ
          END IF

320     CONTINUE

      END IF


      RETURN

      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       QNTLAR (NLEV, VMIN, VMAX, XQ, YQ)
*
*
* PURPOSE:
*       THIS ROUTINE GENERATES QUANTIZER TABLES FOR THE LOG-AREA
*       QUANTIZATION OF REFLECTION COEFFICIENTS.
*
*
* DESCRIPTION:
*       THIS ROUTINE GENERATES A TABLE OF QUANTIZER BREAK POINTS
*       AND QUANTIZER OUTPUT LEVELS FOR QUANTIZING REFLECTION
*       COEFFICIENTS. THE QUANTIZER HAS THE SAME CHARACTERISTICS
*       AS ONE WHICH EMPLOYS A LOG-AREA RATIO NON-LINEARITY.
*
*
* PARAMETERS:
*       NLEV    - NUMBER OF QUANTIZER LEVELS
*       VMIN    - MINIMUM VALUE FOR THE REFLECTION COEFFICIENTS
*                 (-1 < VMIN < VMAX < +1)
*       VMAX    - MAXIMUM VALUE FOR THE REFLECTION COEFFICIENTS
*   (*) XQ      - OUTPUT VECTOR OF NLEV-1 QUANTIZER BREAK POINTS
*   (*) YQ      - OUTPUT VECTOR OF NLEV QUANTIZER OUTPUT LEVELS
*
*
* ROUTINES REQUIRED:
*       NONE
*
*
* AUTHOR / MAINTAINED BY:
*       P. Kabal
*
*
* DATE CREATED:
*       80/07/09
*
*
* UPDATES:
*       80/07/23
*       80/08/06
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
      SUBROUTINE QNTLAR (NLEV, VMIN, VMAX, XQ, YQ)

      REAL XQ(NLEV-1),YQ(NLEV)

*  TRANSFORM THE REFLECTION COEFFICIENT RANGE TO
*  A LOG-AREA RATIO RANGE
      XMAX=LOG((1.0+VMAX)/(1.0-VMAX))
      XMIN=LOG((1.0+VMIN)/(1.0-VMIN))
      SCL=0.5*(XMAX-XMIN)
      OFF=0.5*(XMAX+XMIN)

*  GENERATE THE QUANTIZER OUTPUT LEVELS
      DO 100 I=1,NLEV
        YY=EXP( (2*I-(NLEV+1)) *SCL/NLEV +OFF)
        YQ(I)=(YY-1.)/(YY+1.)
100   CONTINUE

* GENERATE THE QUANTIZER BREAK POINTS
      DO 200 I=1,NLEV-1
        YY=EXP( (2*I-NLEV) *SCL/NLEV +OFF)
        XQ(I)=(YY-1.)/(YY+1.)
200   CONTINUE

      RETURN

      END
```

```
*---------------bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       SUBROUTINE QNTLOG (NLEV, AMU, VMIN, VMAX, XQ, YQ)
*
*
* PURPOSE:
*       This routine generates quantizer tables for the pseudo-
*       logarithmic quantization of parameter values.
*
*
* DESCRIPTION:
*       This routine generates a table of quantizer break points
*       and quantizer output levels for quantizing parameters.
*       The quantizer has the same characteristics as one which
*       employs a pseudo-logarithmic non-linearity.
*
*
* PARAMETERS:
*       NLEV    - Number of quantizer levels
*       AMU     - Quantizer parameter (AMU > 0 ). When AMU is very
*                 small, the quantizer is nearly uniformly spaced.
*       VMIN    - Minimum value for the parameters to be quantized
*                 (-1 < VMIN < VMAX < +1)
*       VMAX    - Maximum value for the parameters to be quantized
*    (*) XQ     - output vector of NLEV-1 quantizer break points
*    (*) YQ     - Output vector of NLEV quantizer output levels
*
*
* ROUTINES REQUIRED:
*       ULAW    - Inverse mu-law function
*
*
* AUTHOR / MAINTAINED BY:
*       P. Kabal
*
*
* DATE CREATED:
*       81/07/03
*
*
* UPDATES:
*
*
*---------------bell-Northern Research / INRS Computer Laboratory-------------

      SUBROUTINE QNTLOG (NLEV, AMU, VMIN, VMAX, XQ, YQ)

      REAL XQ(NLEV-1),YQ(NLEV)

*  Generate the quantizer output levels
      DO 100 I=1,NLEV
        YY=FLOAT(2*I-1)/(2*NLEV)
        YQ(I)=(VMAX-VMIN)*ULAW(YY,AMU)+VMIN
100   CONTINUE

* Generate the quantizer break points
      DO 200 I=1,NLEV-1
        YY=FLOAT(I)/NLEV
        XQ(I)=(VMAX-VMIN)*ULAW(YY,AMU)+VMIN
200   CONTINUE

      RETURN

      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       QUANTZ (X, L, XQ, NLEV)
*
*
* PURPOSE:
*       THIS ROUTINE QUANTIZES AN INPUT VALUE.
*
*
* DESCRIPTION:
*       THIS SUBROUTINE RETURNS THE INDEX OF THE QUANTIZER OUTPUT REGION
*       CORRESPONDING TO A GIVEN INPUT VALUE. THE QUANTIZER IS SPECIFIED
*       BY AN ARRAY OF QUANTIZER BREAK POINTS.
*
*
* PARAMETERS:
*       X       - QUANTIZER INPUT VALUE
*    (*) L      - OUTPUT INDEX OF THE QUANTIZER OUTPUT REGION CORRESPONDING
*                 TO X. L TAKES ON VALUES FROM 1 TO NLEV.
*                    L = 1,        X =< XQ(1)
*                    L = 2,        XQ(1) < X =< XQ(2)
*                     ...           ...
*                    L = NLEV,     XQ(NLEV-1) < X
*       XQ      - INPUT ARRAY OF NLEV-1 QUANTIZER BREAK POINTS WHICH DELIMIT
*                 THE NLEV QUANTIZER REGIONS. THESE MUST BE IN ASCENDING ORDER.
*       NLEV    - NUMBER OF QUANTIZER REGIONS (MINIMUM 2)
*
*
* ROUTINES REQUIRED:
*       NONE
*
*
* AUTHOR / MAINTAINED BY:
*       P. KABAL
*
*
* DATE CREATED:
*       79/05/24
*
*
* UPDATES:
*       80/06/17
*
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
```

```
      SUBROUTINE QUANTZ (X, L, XQ, NLEV)

      REAL XQ(NLEV-1)

*  BINARY SEARCH FOR THE INTERVAL ( XQ(IL),XQ(IU) ) WHICH BRACKETS
*  THE VALUE X.
      IL=0
      IU=NLEV

100   I=(IL+IU)/2
      IF(X.LE.XQ(I)) THEN
        IU=I
      ELSE
        IL=I
      ENDIF

*  SEE IF THE INTERVAL HAS BEEN ISOLATED
      IF(IU.GT.IL+1) GO TO 100

      L=IU

      RETURN

      END
```

```
C----------Bell-Northern Research / INRS Computer Laboratory----------
C
C
C MODULE:   SUBROUTINE RANGE
C
C
C FILE NAME:  RANGE (N,IW,LW,CI,CO)
C
C
C PURPOSE: RANGER EN ORDRE DES ELEMENTS DES TABLEAUX IW,LW,CI,CO
C
C
C DESCRIPTION:  RANGE PETIT DES ELEMENTS EN ORDRE CROISSANT EN SE
C               BASANT SUR DES ELEMENTS DE IW
C
C
C PARAMETERS:
C               N = DIMENSION DES TABLEAUX
C               IW,LW,CI,CO = 4 TABLEAUX
C
C
C ROUTINES REQUISES: AUCUNE
C
C
C SPECIAL REQUIREMENTS:
C
C
C SIDE EFFECTS:
C
C
C AUTHOR / MAINTAINED BY:  C. SIDE
C
C
C DATE CREATED:  6-12-81
C
C
C UPDATES:
C
C----------Bell-Northern Research / INRS Computer Laboratory----------
```

```
      SUBROUTINE RANGE (N,IW,LW,CI,CO)
C
      DIMENSION IW(N),LW(N),CO(N),CI(N)
C
      DO 10 I=1,N-1
C
      IF(IW(I+1).LT.IW(I))THEN
C
       J=I
100    J=J-1
       IF(J.LE.0)GO TO 200
       IF(IW(I+1).LT.IW(J))GO TO 100
C
200    J=J+1
       A=IW(I+1)
       B=LW(I+1)
       C=CI(I+1)
       D=CO(I+1)
C
       DO 20 K=I,J,-1
       IW(K+1)=IW(K)
       LW(K+1)=LW(K)
       CI(K+1)=CI(K)
20     CO(K+1)=CO(K)
C
       IW(J)=A
       LW(J)=B
       CI(J)=C
       CO(J)=D
C
      END IF
10    CONTINUE
C
      RETURN
      END
```

```
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
*
*
* MODULE:
*       SUBROUTINE WNDGEN (WIN, NPTS, A)
*
*
* PURPOSE:
*       This routine generates a raised cosine window.
*
*
* DESCRIPTION:
*       The window satisfies the following equation.
*
*       WIN(I)=(1-A)-A*COS((I-1)*2*PI/(NPTS-1))
*
*           A = 0.46 for a Hamming window,
*           A = 0.5  for a full raised cosine window.
*
*
* PARAMETERS:
*   (*) WIN   - Vector containing the window
*       NPTS  - Number of points in the window
*       A     - Parameter for the window
*
*
* ROUTINES REQUIRED:
*       None
*
*
* AUTHOR / MAINTAINED BY:
*       P. Kabal
*
*
* DATE CREATED:
*       15-Apr-80
*
*
*---------------Bell-Northern Research / INRS Computer Laboratory-------------
```

```
      SUBROUTINE WNDGEN (WIN, NPTS, A)

      PARAMETER (PI=3.14159265)

      REAL WIN(0:NPTS-1)


      F=2.*PI/(NPTS-1)

      DO 100 I=0,NPTS-1
        WIN(I)=(1.-A)-A*COS(I*F)
100   CONTINUE


      RETURN

      END
```