88/174

Sc N 88/138 A

Université de Nancy I — U.E.R. Sciences mathématiques

Centre de Recherche en Informatique de Nancy

CODAGE ET RECONNAISSANCE DE LA PAROLE PAR QUANTIFICATION VECTORIELLE

INSTITUT POLYTECHNIQUE DE LORRAINE NANCY

THESE

Présentée et Soutenue Publiquement le 1er Avril 1988
à l' Université de Nancy I
pour l'obtention du titre de

DOCTEUR DE L'UNIVERSITE DE NANCY I
EN INFORMATIQUE

par

Ahmed GOURINDA

Composition du Jury:

Président:	R.	Mohr
Rapporteurs:	R.	Carré
	R.	Schott
Examinateurs:	J.P.	Haton
	J.M.	Pierrel

Université de Nancy I U.E.R. Sciences mathématiques

Centre de Recherche en Informatique de Nancy

CODAGE ET RECONNAISSANCE DE LA PAROLE PAR QUANTIFICATION VECTORIELLE

THESE

Présentée et Soutenue Publiquement le 1er Avril 1988
à l' Université de Nancy I
pour l'obtention du titre de

DOCTEUR DE L'UNIVERSITE DE NANCY I
EN INFORMATIQUE
par

Ahmed GOURINDA

Composition du Jury:

Président:	R. Mohr
Rapporteurs:	R. Carré
	R. Schott
Examinateurs:	J.P. Haton
	J.M. Pierrel

Je tiens à exprimer ma profonde gratitude à Mr le professeur Haton qui a bien voulu m'accueillir dans son laboratoire. Je lui suis infiniment redevable d'avoir su inspirer et diriger ce travail.

Je remercie Messieurs les Professeurs Carré et Schott qui ont bien voulu juger ce travail malgré l'importance des nombreuses tâches qu'ils assument.

Je remercie également le Professeur Pierrel pour avoir accepté de participer à ce jury. Le professeur Mohr me fait un grand honneur en acceptant de présider ce jury.

Tous mes remerciements vont également à Joseph Di Martino et François Charpillet pour leurs conseils et critiques qui m'ont permis d'améliorer ce travail.

Merci enfin à tous mes camarades du CRIN pour l'aide précieuse et amicale qu'ils m'ont apportée lors de la réalisation de ce travail.

Table des matières

Chapitre 1

INTRODUCTION

La parole constitue sans aucun doute le mode de communication le plus naturel pour l'homme. Les systèmes traditionnels d'interfaçage homme-machine (clavier avec touches programmables, souris, mémorisation automatique des commandes précédentes, etc) ne permettent qu'une interaction limitée avec un environnement informatique et sont loin d'égaler le confort et l'universalité que présente un dialogue oral. Celui-ci est d'autant plus riche que le signal vocal est porteur non seulement du message sémantique, objet de la communication, mais encore l'information sur le locuteur (sexe, âge, l'accent régional, etc.).

Un des buts à atteindre pour les nombreux chercheurs qui travaillent dans le domaine de la parole (reconnaissance, dialogue, etc.) est de faciliter le dialogue homme-machine : à savoir pouvoir commander par la voix une machine à écrire, un robot, interroger une base de données ou un centre de renseignements.

Ce but est, comme on le verra tout au long de ce travail, difficile à atteindre avec les moyens actuels. En parole, il est très difficile d'extraire des paramètres intrinsèques. En effet ceux-ci varient d'un locuteur à l'autre. Néanmoins, pour des applications spécifiques comprenant un nombre limité de mots, une syntaxe rigide et peu de locuteurs, il existe des systèmes capables de fonctionner avec une fiabilité suffisante [10].

Les progrès des techniques de traitement du signal et l'apport des méthodes d'adaptation au locuteur ainsi que celui de l'intelligence artificielle permettent d'envisager à l'avenir de véritables systèmes de dialogue homme-machine.

Cette étude, consistant à appliquer la quantification vectorielle aux divers domaines de la parole comporte quatre parties (figure 1.1).

- Dans le deuxième chapitre nous présentons un rappel détaillé des techniques usuelles de paramétrisation du signal de parole. Nous allons surtout insister d'une part sur le fondement mathématique de ces techniques et d'autre part sur le lien fort intéressant existant entre le signal analogique et le signal numérique.

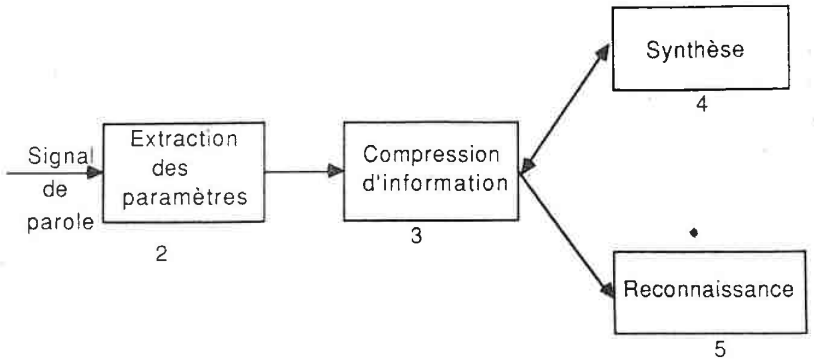


Figure 1.1: Plan de la thèse

- Dans le troisième chapitre nous abordons les différents problèmes de compression d'information. Nous allons surtout insister sur l'apport de la quantification vectorielle. Nous allons étudier les différents algorithmes de génération de prototypes. Nous terminons ce chapitre en proposant un algorithme d'apprentissage qui tient compte des problèmes de stabilité des filtres.

- La synthèse de la parole fait l'objet du quatrième chapitre. Nous utilisons la synthèse pour montrer la validité de la quantification vectorielle. Nous proposons un synthétiseur réalisant un compromis entre le débit de transmission et la qualité d'écoute.

- Le cinquième chapitre est consacré à la reconnaissance de la parole. Nous allons comparer plusieurs techniques de reconnaissance. Nous proposons une méthode rapide de reconnaissance de mots isolés, fondée sur la quantification vectorielle multi-sections avec normalisation linéaire variable de chaque mot du vocabulaire et moyennage de toutes les trames de signal constituant une section. Nous raffinons la décision de cette dernière par adjonction d'une deuxième méthode utilisant des formes de références non quantifiées. Nous proposons aussi une extension de notre approche à la reconnaissance de la parole continue, les résultats expérimentaux montrent que notre méthode assure un compromis complexité/performance intéressant.

Chapitre 2

PARAMETRISATION DU SIGNAL VOCAL

1 ACQUISITION DE LA PAROLE

Pour l'utilisateur le mécanisme est très simple, il suffit de parler devant un microphone pour récupérer un signal de parole digitalisée dans un fichier.

Parmi les possibilités qu'offre la procédure d'acquisition est le choix de la fréquence d'échantillonnage. Cette fréquence permet de déterminer le nombre de points qu'on veut obtenir pendant une seconde de parole. Si par exemple on choisit 16 Khz comme fréquence d'échantillonnage on aura 16000 points/seconde. Comme chaque point pour le cas de MASSCOMP est codé sur 12 bits (une dynamique entre -2048 et 2047) on aura un débit de 192000 bits/s.

Quand on augmente la fréquence d'échantillonnage (en diminuant Δt) l'extraction du signal digitalisé se fait avec plus de précision (figure 2.1), mais on augmente la taille du signal recupéré, ce qui pour un traitement ultérieur demande plus de temps.

Une fréquence de 16 Khz fournit une bande suffisante (figure 2.2) pour l'étude des différents sons de parole. Au-delà de cette fréquence beaucoup d'informations deviennent redondantes, et de plus le temps de traitement augmente considérablement.

2 RESTITUTION DE LA PAROLE

Grâce aux moyens informatiques, les techniques numériques se développent de plus en plus et permettent de traiter numériquement tous les signaux analogiques.

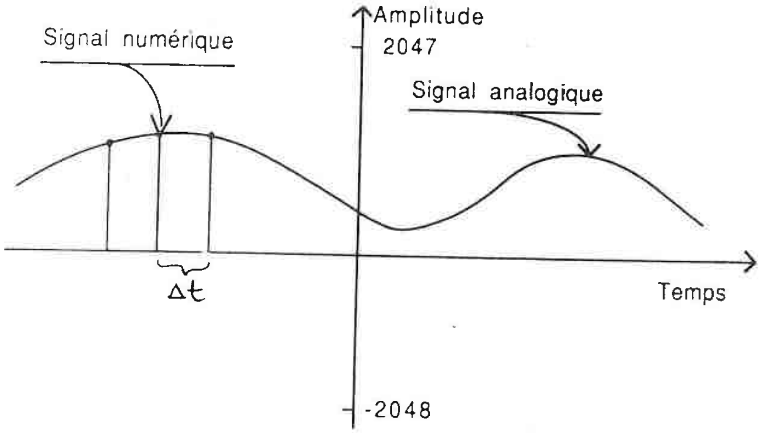


Figure 2.1: $\frac{1}{\Delta t}$ représente la fréquence d'échantillonnage.

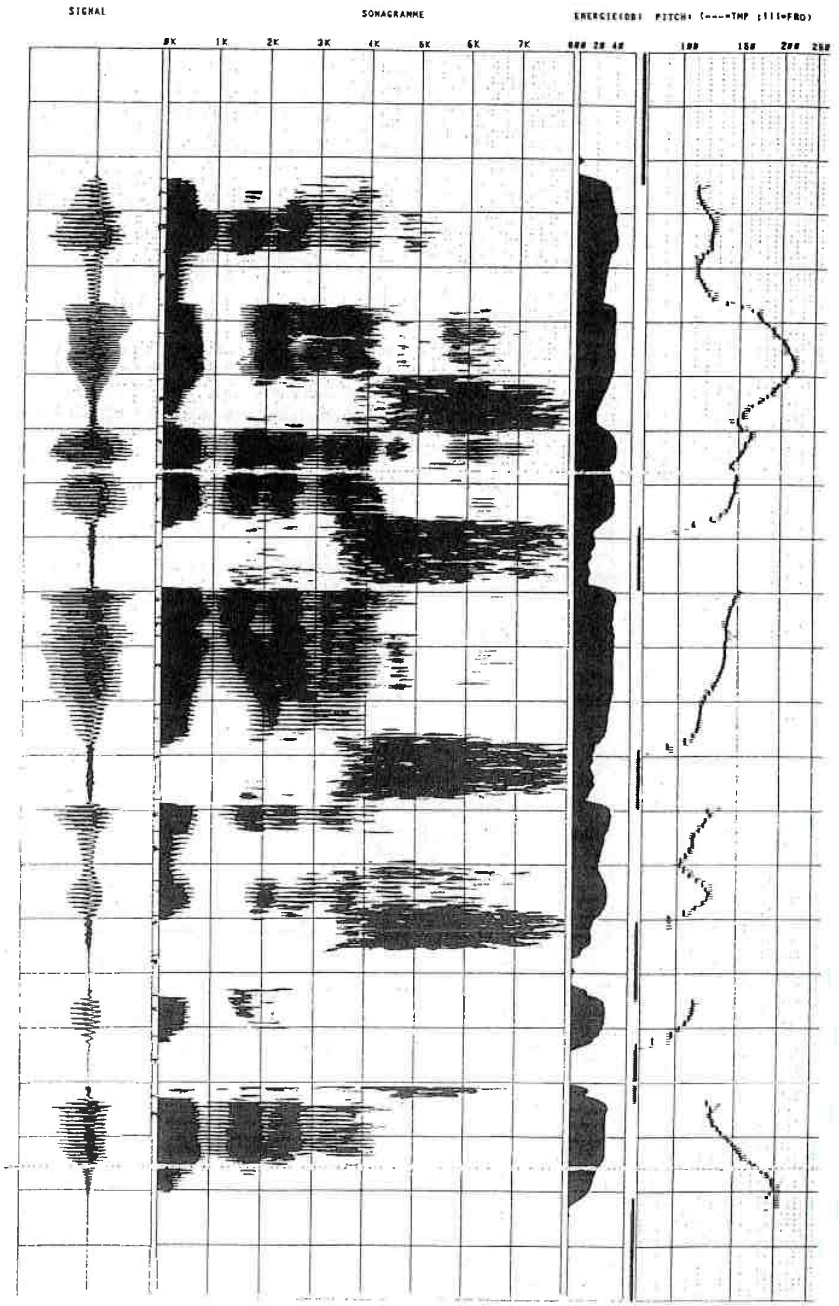


Figure 2.2: Sonagramme de "La bise et le soleil" (< *LABISA.FL* > CNET LANNION)

La restitution est l'opération qui permet de retrouver à partir d'un signal numérique le signal analogique de départ. Cette transformation est possible si on respecte certaines conditions.

3 TRANSFORMEE DE FOURIER DISCRETE (TFD)

En traitement du signal une technique qui joue un rôle primordial est la transformée de Fourier.

Pratiquement quand on analyse un signal on ne le traite que fenêtre (ou trame) par fenêtre. A un moment donné on ne connait que les observations x(0), x(1),....., x(N-1). N étant la longueur de la fenêtre temporelle d'analyse. Pour pouvoir appliquer la théorie de Fourier on suppose que cette fenêtre se répète indéfiniment dans le temps, donc on obtient un signal périodique de période N.

3.1 DEFINITIONS

L'expression mathématique de la TFD est donnée par :

$$X(n) = \sum_{k=0}^{N-1} x(k) exp(-j2\pi \frac{nk}{N}) \qquad (2.1)$$

avec n=0,.....,N-1 et la TFD inverse par :

$$x(k) = \frac{1}{N} \sum_{n=0}^{N-1} X(n) exp(j2\pi \frac{nk}{N})$$

3.2 PROPRIETES

x(k) = x(k + iN) quelque soit i entier

X(n) = X(n + iN) quelque soit i entier

X(n - N/2) = X(n + N/2) avec n=0, ... ,N-1

3.3 TRANSFORMEE DE FOURIER RAPIDE (FFT)

La transformée de Fourier rapide (FFT) n'est qu'un moyen rapide, élégant et économique pour calculer la TFD. Rappelons que le principe de la FFT (Fast Fourier Transform) est de retarder jusqu'au dernier moment le calcul inévitable de la TFD.

On a souvent recours à la FFT pour l'analyse spectrale.

4 SPECTRE D'ENERGIE

L'expression mathématique du spectre d'énergie est donnée par : $|X(n)|^2$, X(n) étant la transformée de Fourier (2.1) du signal temporel x(n). Le spectre d'energie exprime la répartition fréquentielle de l'énergie du signal.

4.1 PROPRIETES DU SPECTRE D'ENERGIE

$$|X(-k)|^2 = |X(k)|^2$$

$$|X(N/2-k)|^2 = |X(N/2+k)|^2 \qquad (2.2)$$

La relation (2.2) exprime le fait que le spectre d'énergie est symétrique par rapport à N/2. La zone fréquentielle porteuse d'informations est la zone comprise entre 0 et fs/2. (fs étant la fréquence d'échantillonnage du signal).

La grande question qui se pose est de savoir si la transformation analogique-numérique est sans danger et s'il y a des conditions à respecter pour passer d'un signal analogique à un signal numérique et vice-versa.

5 THEOREME D'ECHANTILLONNAGE

Theorème [26,40] :
Un signal analogique $x_a(t)$ ayant une largeur de bande finie limitée à F(Hz) (figure 2.3) ne peut être reconstitué exactement à partir de ses échantillons $x(k\Delta t)$ que si ceux-ci ont été prélevés avec une période Δt inférieure ou égale à 1/(2F).

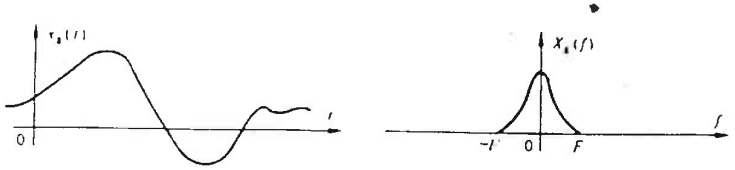


Figure 2.3: Signal temporel continu et son spectre

Ce théorème est d'une importance capitale, il permet d'établir le lien entre le traitement numérique et le traitement analogique. Ainsi on peut effectuer numériquement des opérations difficiles, voire impossibles par voie analogique.

L'échantillonnage est réalisé en multipliant le signal analogique $x_a(t)$ par une suite périodique d'impulsions de Dirac de période Δt.

$$x_e(t) = x_a(t).e(t)$$

avec :

$$e(t) = \delta_\Delta(t)$$

La transformée de Fourier $X_e(f)$ du signal échantillonné est donnée par :

$$X_e(f) = X_a(f) \star E(f) = \frac{1}{\Delta t} \sum_{n=-\infty}^{n=+\infty} X_a(f - \frac{n}{\Delta t})$$

L'échantillonnage (figure 2.4b) fait apparaître une succession de spectres proportionnels au spectre du signal analogique.

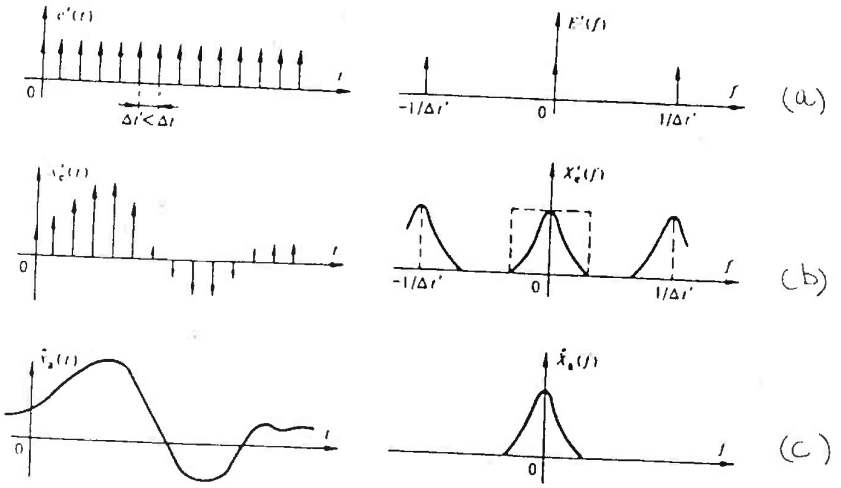


Figure 2.4: Spectre du signal échantillonné

Deux cas peuvent se présenter :

1. $\frac{1}{2\Delta t} < F$

On remarque d'après la figure 2.4b qu'il y a un chevauchement des spectres notamment autour des points $\frac{k}{2\Delta t}$. Pour retrouver le signal analogique il faut essayer de retrouver le spectre de la figure 2.3, il faut donc éliminer par un filtre passe-bas idéal de fréquence de coupure $\frac{1}{2\Delta t}$ toutes les fréquences au dessus de la fréquence de coupure. Or même avec un filtre idéal on ne retrouve pas le spectre de la figure 2.4 à cause du chevauchement

des spectres (figure 2.4c). Le signal estimé $\hat{x}_a(t)$ est alors différent du signal de départ.

2. $\Delta t' < \Delta t$ *et* $\frac{1}{2\Delta t'} \geq F$

Dans ce cas on n'a plus de chevauchement des spectres(figure 2.5b). Par filtrage passe-bas idéal on isole le spectre de la figure 2.3. Ainsi aux erreurs près dûes au filtrage on retrouve le signal analogique de départ (figure 2.5c).

Notons au passage que le filtre passe-bas idéal n'est pas causal (chapitre 4), il n'est par conséquent pas réalisable, donc on recourt à d'autres techniques d'interpolations pour le réaliser.

6 CODAGE PAR PREDICTION LINEAIRE

Le signal temporel vocal est quasi périodique et assez régulier, surtout dans les regions voisées. Il se prête assez bien pour la prédiction linéaire. Dans les zones non voisées, caractérisées par un bruit aléatoire la prédiction est moins fiable, mais n'altère pas globalement l'efficacité de la méthode.

Le principe de cette technique [33] est de prédire la valeur du signal à l'instant n connaissant les p observations précédentes (figure 2.6). Autrement dit on cherche un modèle qui approche le mieux ce signal. La méthode fournit à la fois une paramétrisation robuste et une compression importante du signal vocal.

Le terme (LPC) ou codage par prédiction linéaire contient deux mots-clés : prédiction et linéarité.

6.1 PREDICTION D'ORDRE p

Elle exprime le fait que $\hat{x}(n)$ est une combinaison quelconque de x(n-p), ... ,x(n-1) :

$$\hat{x}(n) = f(x(n-p), ..., x(n-1))$$

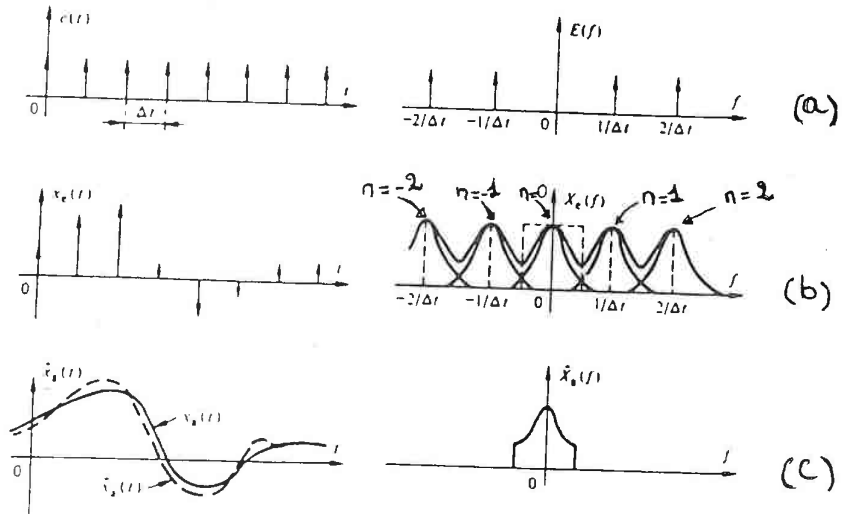


Figure 2.5: Reconstitution du signal initial

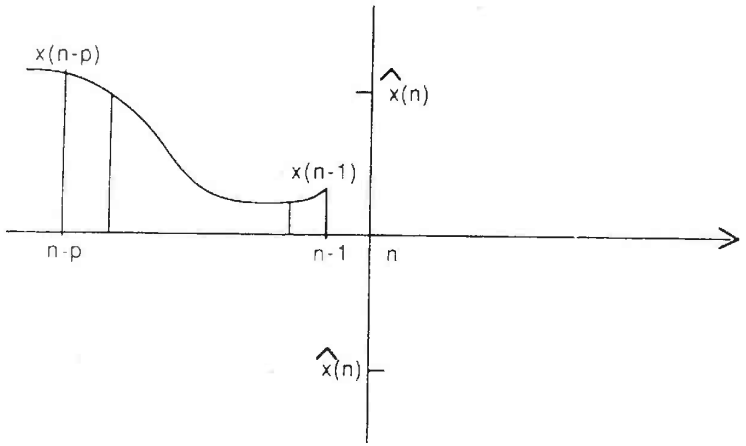


Figure 2.6: A partir des observations du signal aux instants n-1, ... ,n-p on veut prédire la valeur $\hat{x}(n)$ du signal à l'instant n. $\hat{x}(n)$ peut prendre toutes les valeurs sur l'axe vertical

6.2 LINEARITE

Cette propriété permet de restreindre le domaine des fonctions f à celui des fonctions lineaires, donc :

$$\hat{x}(n) = b_1 x(n-1) + b_2 x(n-2) + ... + b_p x(n-p)$$

On pose $a_i = -b_i$ la dernière relation devient :

$$\hat{x}(n) = -a_1 x(n-1) - a_2 x(n-2) - .... - a_p x(n-p)$$

Qu'on peut écrire encore :

$$\hat{x}(n) = -\sum_{i=1}^{p} a_i x(n-i)$$

Comme on connait x(n) on choisit les a_i de manière à ce que la prédiction $\hat{x}(n)$ soit la plus proche possible de x(n). On définit l'erreur e(n) par :

$$e(n) = x(n) - \hat{x}(n) = x(n) + \sum_{i=1}^{p} a_i x(n-i)$$

$$= \sum_{i=0}^{p} a_i x(n-i) \qquad avec\ :\ a_0 = 1$$

Cette dernière relation peut s'écrire encore :

$$x(n) = -\sum_{i=1}^{p} a_i x(n-i) + e(n) \tag{2.3}$$

Le bon prédicteur est celui qui minimise l'erreur global E (critère des moindres carrés) définie par :

$$E = \sum_n e^2(n)$$

$$= \sum_n (\sum_{i=0}^{p} a_i x(n-i))(\sum_{j=0}^{p} a_j x(n-j))$$

$$= \sum_{i=0}^{p}\sum_{j=0}^{p} a_i [\sum_n x(n-i)x(n-j)]a_j \qquad (2.4)$$

On pose

$$C_{ij} = \sum_n x(n-i)x(n-j) \qquad (2.5)$$

La relation (2.4) devient :

$$E = \sum_{i=0}^{p}\sum_{j=0}^{p} a_i C_{ij} a_j \qquad (2.6)$$

Minimiser l'erreur E revient à résoudre l'équation $\frac{\partial E}{\partial a_j} = 0$ pour j=1, ... ,p, ce qui revient à résoudre le système :

$$\sum_{i=0}^{p} a_i C_{ij} = 0 \quad pour \quad j = 1, ..., p$$

Ce qui peut s'écrire encore :

$$\sum_{i=1}^{p} a_i C_{ij} = -C_{0j} \qquad (2.7)$$

6.3 CALCUL DES C_{ij}

Dans la relation (2.5) l'intervalle de variation de n n'a pas été précisé. Il y a deux méthodes pour résoudre le système (2.7) selon les valeurs prises par n : la méthode d'autocorrélation et la méthode de covariance.

6.4 METHODE D'AUTOCORRELATION

Le domaine de variation de n est alors $]-\infty, +\infty[$. Comme on travaille sur une fenêtre de longueur N on ne connaît que les observations x(0), ... , x(N-1). Par définition les coefficients d'autocorrélation correspondants à ce signal sont donnés par :

$$r(i) = \sum_{n=0}^{N-i-1} x(n)x(n+i)$$

L'équation (2.5) s'ecrit :

$$C_{ij} = \sum_{n=-\infty}^{+\infty} x(n-i)x(n-j)$$

$$= \sum_{n=-\infty}^{+\infty} x(n)x(n+|i-j|)$$

$$= \sum_{n=0}^{N-1-|i-j|} x(n)x(n+|i-j|)$$

$$= r(|i-j|)$$

L'équation (2.7) devient :

$$\sum_{i=1}^{p} a_i r(|i-j|) = -r(j) \qquad pour \quad j = 1, ..., p \qquad (2.8)$$

Ou encore :

$$\sum_{i=0}^{p} a_i r(|i-j|) = 0 \qquad pour \;\; j = 1, ..., p$$

L'écriture matricielle du système (2.8) fait apparaître une matrice symétrique ayant le même élément sur chaque diagonale (matrice de type Toeplitz). Pour la résolution de ce système il existe une méthode rapide, exploitant toutes les propriétés de cette matrice particulière, élaborée par Durbin/Levinson. Cette méthode consiste à transformer le système (2.8) en relations récurrentes :

$$E_0 = r(0)$$

$$k_i = -[r(i) + \sum_{j=1}^{i-1} a_j^{(i-1)} r(i-j)]/E_{i-j}$$

$$a_j^{(i)} = k_i$$

$$a_j^{(i)} = a_j^{(i-1)} + k_i a_{i-j}^{(i-1)}. \quad pour \;\; 1 \leq j \leq i-1$$

$$E_i = (1 - k_i^2) E_{i-1}$$

Les quatre dernières équations sont évaluées récursivement pour $i = 1, ..., p$. La solution finale est donnée par :

$$a_j = a_j^{(p)}, \quad 1 \leq j \leq p$$

Ainsi les coefficients LPC sont obtenus par une procédure récursive. Il est à noter que la méthode d'autocorrélation est la méthode la plus utilisée, car les coefficients qu'on obtient sont toujours stables [33].

Après détermination des coefficients a_i on remplace les C_{ij} par leurs valeurs dans l'equation (2.6). On obtient alors l'erreur de prédiction qui est donnée par :

$$E_m = r(0) + \sum_{i=1}^{p} a_i r(i) \tag{2.9}$$

En combinant le système (2.8) et cette dernière équation on obtient la forme matricielle suivante :

$$\begin{vmatrix} r(0) & r(1) & r(p-1) & r(p) \\ r(1) & r(0) & r(p-2) & r(p-1) \\ & & & \\ & & & \\ r(p-1) & r(p-2) & r(0) & r(1) \\ r(p) & r(p-1) & r(1) & r(0) \end{vmatrix} \cdot \begin{vmatrix} 1 \\ a_1 \\ \\ \\ a_{p-1} \\ a_p \end{vmatrix} = \begin{vmatrix} E_m \\ 0 \\ \\ \\ 0 \\ 0 \end{vmatrix}$$

On remarque que :

$$a' R a = E_m$$

où :

R est la matrice des autocorrélations du signal
a = (1,a_1, ... ,a_{p-1}, a_p)' cofficients LPC
' désigne l'opération de transposition

Cette forme matricielle sera souvent utilisée pour le calcul des distances (chapitre 3).

6.5 METHODE DE COVARIANCE

Pour cette méthode n $\in$ [p, N[donc :

$$C_{ij} = \sum_{n=p}^{N-1} x(n-i) x(n-j)$$

On utilise la décomposition de Cholesky pour résoudre le système (2.7). Cette méthode, bien que plus précise que la méthode d'autocorrélation surtout pour les fenêtres de courte durée est rarement utilisée car le filtre qu'on obtient n'est pas toujours stable. Il existe des méthodes mathématiques pour transformer tout filtre instable en un filtre stable, mais le coût de l'opération est trop élevé (il faut soit ajouter un petit nombre à tous les termes diagonaux de la matrice des covariances soit, de façon rigoureuse chercher toutes les racines de l'équation $1+\sum_{i=1}^{p} a_i z^{-i} = 0$ dont le module est supérieur ou égal à 1, puis les remettre à l'intérieur du cercle unité).

6.6 ANALYSE SPECTRALE ET LPC

Le spectre du modèle autorégressif obtenu par prédiction linéaire(LPC) représente une sorte de lissage du spectre de la FFT (figure 2.7). La régularité de ce spectre fait de lui un moyen efficace de comparaison des modèles (chapitre 3).

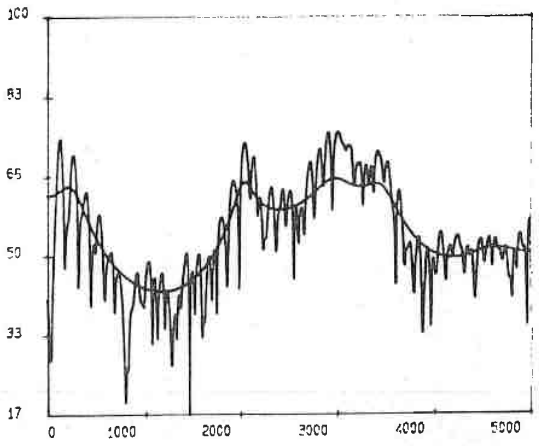


Figure 2.7: Le spectre de la FFT et le spectre de LPC (une sorte de lissage du spectre de la FFT) d'un même signal échantillonné à 10 Khz

7 ANALYSE PAR BANC DE FILTRES

Le principe de cette technique d'analyse est de répartir l'énergie du signal vocal selon des bandes de fréquences, ou canaux. Chaque canal est réalisé par un filtre passe-bande. Ces filtres peuvent avoir plusieurs formes (triangulaire, gaussienne, etc.), et ils sont disposés suivant différentes échelles. L'échelle couramment utilisée et qui tient compte de la perception humaine est l'échelle Mel qui est linéaire jusqu'à 1000 Hz, puis logarithmique (figure 2.8).

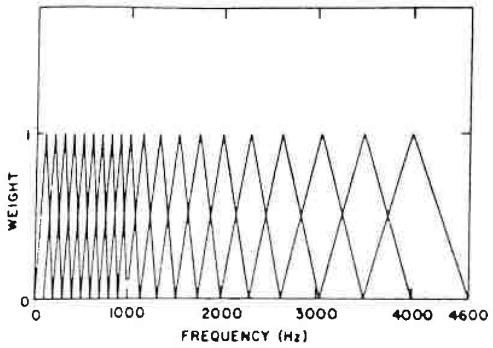


Figure 2.8: exemple de filtres pour l'extraction des coefficients MFCC. Ces filtres sont disposés selon l'échelle Mel

On obtient différentes paramétrisations selon la manière dont on combine les sorties des filtres. Par exemple des paramétrisations dérivées de la FFT (MFCC, LFCC, etc.) [34,9]. Puisque la sortie de chaque canal est une mesure exacte de l'énergie du signal dans une bande de fréquence, les coefficients MFCC, par exemple sont plus robustes que les coefficients LPC dans les zones bruitées.

8 COEFFICIENTS DE REFLEXION

Les coefficients de réflexion (k_i) sont obtenus par transformation (2.10) non linéaire des coefficients a_i de LPC. Parmi les avantages de ces coefficients on peut citer :

1. Ils sont compris strictement entre -1 et 1.

2. Les premiers coefficients sont inchangés lors du passage d'un ordre vers un ordre supérieur :

   si p' > p alors $k_i = k'_i$ pour i=1, ...,p
   $(k_1, k_2, ..., k_p)$ les coefficients de réflexion d'ordre p et
   $(k'_1, k'_2, ..., k'_{p'})$ les coefficients de réflexion d'ordre p'

3. Tout coefficient supérieur ou égal à 1 suffit pour détecter l'instabilité du filtre [33].

$$a_{m-1,i} = \frac{a_{mi} - k_m a_{m,m-i}}{1 - k_m^2}$$

$$k_m = a_{mm} \qquad (2.10)$$

$$m = p, p-1, ..., 1 \quad et \quad i = 0, 1, ..., m-1$$

$$avec \ : \ a_{p0} = a_0, \ ....., \ a_{pp} = a_p$$

9 CHOIX DE LA FORME DE LA FENETRE

Comme nous l'avons déjà signalé, à un instant de l'analyse on ne connaît qu'un segment de signal. Ceci se traduit mathématiquement par la multiplication du signal temporel par une fenêtre rectangulaire dont la forme est donnée par $W_r(n)$. Le spectre obtenu résulte en fait de la convolution du spectre du signal et du spectre de la fenêtre. Par conséquent on modifie inévitablement le signal de départ. Ce phénomène est connu sous le nom de phenomène de Gibbs. Pour y remédier on utilise des fenêtres qui altèrent le moins le spectre du signal de départ. La fenêtre de Hamming $W_h(n)$ répond au mieux à ce souci, elle est ainsi couramment utilisée. C'est cette fenêtre que nous avons utilisée dans nos expériences.

$$W_r(n) = \begin{cases} 1 & 0 \leq n \leq N-1 \\ 0 & \text{sinon} \end{cases}$$

$$W_h(n) = \begin{cases} 0.54 - 0.46 \ cos(2\pi \frac{n}{N-1}) & 0 \leq n \leq N-1 \\ 0 & \text{sinon} \end{cases}$$

10 CONCLUSION

Toutes les techniques de paramétrisation du signal, qu'elle soient derivées de la FFT comme MFCC, LFCC ou du LPC comme LPCC, Ki ou d'autres que nous n'avons pas exposées ici, s'efforcent d'extraire le maximum d'information et d'en perdre le minimum. Il est difficile de faire un choix parmi ces techniques. L'oreille humaine est sensible à tout ce qui se passe au niveau spectral. Les techniques favorisant ce côté sont ainsi les plus utilisées.

Nous avons decidé de faire le choix du LPC pour les raisons suivantes :

1. Il existe une procédure, non itérative, pour calculer les coefficients LPC.

2. Contrairement aux autres paramétrisations, il existe une distorsion spécifique à LPC. C'est la distorsion d'Itakura-Saito et ses nombreuses variantes.

3. En général un petit nombre de paramètres est suffisant pour caractériser avec précision le spectre du signal.

4. Le spectre du modèle LPC représente une version lissée du spectre de la FFT du signal.

5. La technique de LPC est basée sur la minimisation de l'énergie de l'erreur. Par conséquent les formants sont représentés avec beaucoup de précision, et souvent leur détection se limite à la détection des pics du spectre.

Chapitre 3

QUANTIFICATION VECTORIELLE

1 INTRODUCTION

Pour pouvoir stocker numériquement une grande quantité de signal dans une mémoire de taille raisonnable, il est nécessaire de compresser le signal en le paramétrant. Beaucoup d'applications ne peuvent utiliser que la forme paramétrée du signal (ou vecteur), on verra au chapitre suivant que pour la transmission de la parole, on peut se contenter de transmettre uniquement la forme paramétrée.

Dans d'autres applications, comme celles qui nécessitent beaucoup de données (par exemple les études statistiques), même la forme paramétrée pose le problème du stockage. Il y a aussi des applications où le stockage direct de la forme paramétrée, même quand elle tient en mémoire, n'est pas souhaité car il nécessite beaucoup de calculs.

La quantification vectorielle est une technique de compression de données qui répond au mieux aux besoins de stockage ou de transmission. la quantification - ou discrétisation - est l'opération qui permet d'approcher un signal à amplitude continue par un signal à amplitude discrète.

La conversion d'un signal analogique (temps continu, amplitude continue) en un signal numérique (temps discret, amplitude discrète) nécessite les deux processus d'échantillonnage et de quantification (figure 3.1). si l'on choisit correctement la fréquence d'échantillonnage (chapitre 2), les seules erreurs possibles sont les erreurs de quantification.

Quand on quantifie séparément chaque composante d'un vecteur, l'opération est connue sous le nom de quantification scalaire alors que quand on quantifie globalement un vecteur on parle de quantification vectorielle (QV).

Il est à noter que la QV peut opérer soit directement sur le signal numérique (suite de

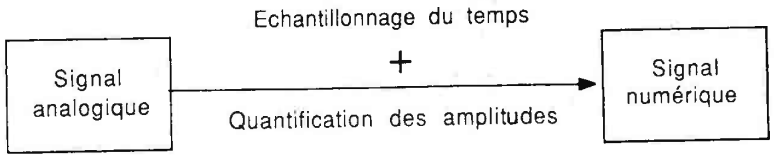


Figure 3.1: transformation analogique-numérique

points qu'on appellera souvent fenêtre) soit, sur le résultat d'une analyse (exemple LPC, MFCC, etc.). C'est alors le vecteur paramétrant la fenêtre d'analyse qui est le vecteur à coder ou à quantifier.

Le mécanisme de quantification scalaire ou vectorielle nécessite l'existence d'une bibliothèque de références qu'on appelera ensemble de prototypes ou de représentants.

Dans ce chapitre on traitera d'abord les différents algorithmes de codage des données par un ensemble de prototypes. On abordera ensuite le problème de génération de ces prototypes. On terminera en proposant un algorithme d'apprentissage.

2 QUANTIFICATION SCALAIRE

Comme le montre la figure 3.2, la quantification scalaire consiste à associer à chaque variable d'entrée x un élément et un seul de l'ensemble de prototypes. Un quantificateur scalaire peut être :

1. Uniforme ou linéaire
   l'axe des x ou l'ensemble des entrées possibles est découpé en intervalles de même longueur. A chaque intervalle est associé un prototype, et le pas de quantification est constant.

2. Non uniforme
   le pas de quantification est alors variable. Certains intervalles recevront plus de prototypes y_i que d'autres. Ce quantificateur est souvent utilisé quand on connait la distribution statistique des données.

Une troisième solution intermédiaire consiste à utiliser un quantificateur linéaire jusqu'à un certain seuil (pour les valeurs équiprobables), puis un quantificateur non uniforme pour le reste.

Pour l'exemple de la figure 3.2, l'ensemble Y des prototypes est :

$$Y = \{y0, y1, y2, y3. y4\}$$

On peut écrire la fonction de quantification sous la forme suivante :

$$y_i = Q(x) \qquad \text{où } i \in [0..4].$$

Un quantificateur fournit une seule information en sortie, mais celle-ci peut-être disponible sous deux formes soit l'indice i, et dans ce cas le i-ème prototype est obtenu par une table de décodage, soit directement le prototype y_i.

3 Application : C.A.N

Il s'agit de décrire sommairement, le principe général de fonctionnement d'un convertisseur analogique numérique (C.A.N) à quatre niveaux (figure 3.3). la sortie de ce quantificateur scalaire uniforme en réponse à chaque entrée est simplement un indice (figure 3.4).

L'ensemble de prototypes qui, dans ce cas correspond à un ensemble d'indices est :

$$Y = \{1, 0, -1, -2\}$$

4 QUANTIFICATION VECTORIELLE

Le principal inconvénient de la quantification scalaire est de ne pas tenir compte de la redondance dans l'information, chaque entrée x étant quantifiée indépendamment des précédentes et des suivantes. La quantification vectorielle, généralisation de la quantification scalaire, permet de résoudre ce problème. cette technique consiste à considérer un vecteur

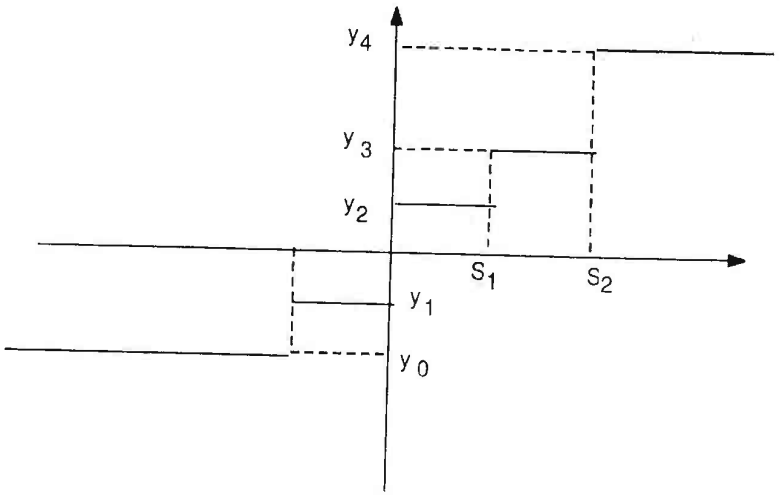


Figure 3.2: exemple de quantificateur scalaire. L'ensemble de prototypes contient y0, y1, y2, y3 et y4. Tous les éléments x compris entre s1 et s2 seront codés par y3

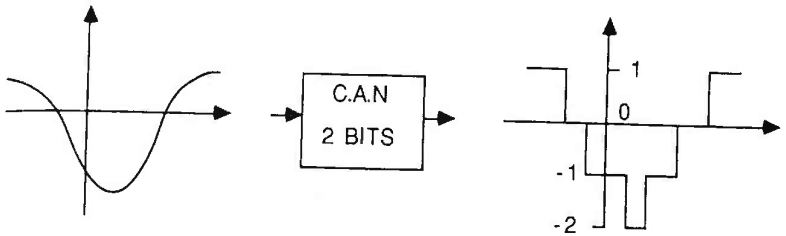


Figure 3.3: schéma général d'un C.A.N

x = (x1,x2....xp) comme un point dans un espace de dimension p, et à coder globalement ce vecteur par un ensemble de prototypes. Ces prototypes réalisent un partitionnement de l'espace en classes (figure 3.5). Ces classes, contrairement à la quantification scalaire, peuvent avoir des tailles et des formes différentes ce qui permet plus de possibilités de quantification. Pour la quantification scalaire (p=1) les classes sont filiformes et définies par des intervalles (figure 3.6).

L'espace étant partitionné, le codage est alors simple : tous les points qui se trouvent à l'intérieur d'une classe C_i recevront le code y_i. Si L'espace est partagé en L classes on aura L prototypes soit :

$$Y = \{y1, y2, \ldots, y_l\}$$

où :

$$y_i = y_{i1}, y_{i2}, \ldots, y_{ip}$$

Y s'appelle l'ensemble de prototypes ou dictionnaire ("codebook") et
y_i prototype ou représentant ("codeword").

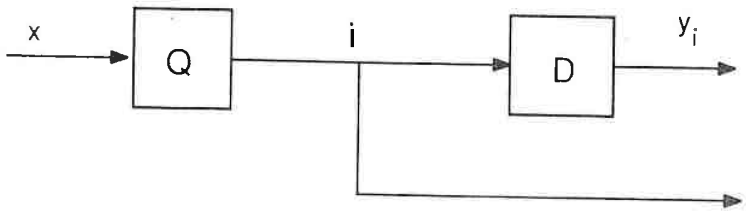


Figure 3.4: principe de quantification d'un convertisseur A/N. On utilise le décodeur D pour retrouver le prototype associé à chaque indice.

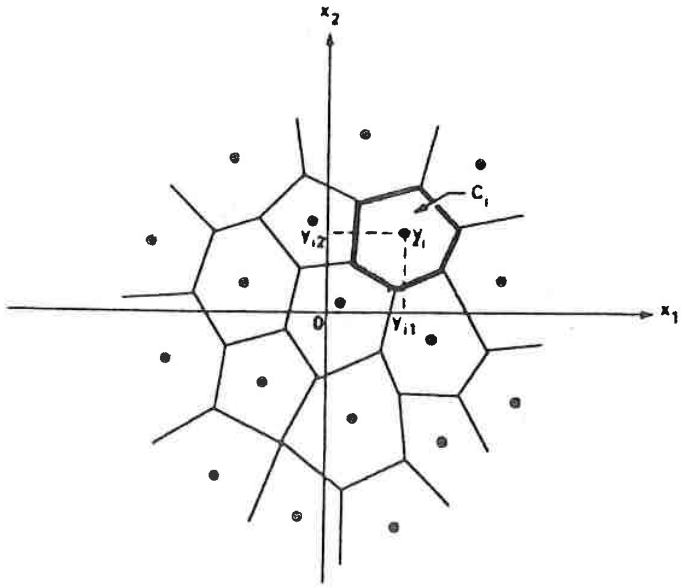


Figure 3.5: l'espace de dimension deux est partitionné en 18 classes. Tous les vecteurs dans la classe C_i seront codés par le prototype y_i . Ces classes ont des formes et des tailles différentes.

On suppose que tous les vecteurs du corpus d'apprentissage appartiennent à un espace de dimension p.

On peut écrire la fonction de quantification de la manière suivante :

$y_i = Q(x) \quad si \quad x \in C_i$

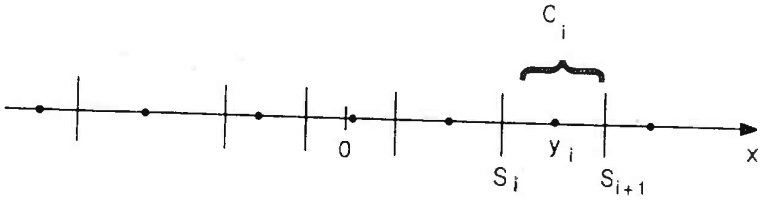


Figure 3.6: quantification scalaire (p=1). les classes ont des tailles différentes mais elles ont la même forme.

4.1 Avantages de la QV

Parmi les avantages de la qv est la possibilité de coder chaque composante d'un vecteur quantifié par une fraction de bit. en effet, si l'ensemble de prototypes contient 2^r éléments, il faudra r bits pour coder le vecteur $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$ et donc $\frac{r}{p}$ bits pour coder x_{ij}. Par ailleurs, la qv reste avantageuse même si les composantes d'un vecteur sont indépendantes [31].

4.2 Quantificateur optimal

Un quantificateur est entièrement déterminé par son ensemble de prototypes et par une distance. En effet, ayant un ensemble de prototypes et une distance la quantification d'un vecteur x par l'ensemble de prototypes y_i est obtenue par application de la règle du plus proche voisin.

En remplaçant x par le code y_i, on introduit forcément une erreur de quantification ou distorsion. Cette erreur peut être mesurée à l'aide d'une distance d. L'information quantifiée est donc plus ou moins bruitée.

Un bon quantificateur est un quantificateur qui altère le moins possible l'information initiale. En général, si on connait la distribution multi-dimensionnelle p(x) pour chaque entrée x, un quantificateur optimal est celui qui minimise l'espérance mathématique de la distance entre l'entrée et la sortie de ce quantificateur.

En pratique ceci revient à minimiser la quantité :

$$D = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} d(x_i, Q(x_i))$$

x_i *et* $Q(x_i)$ sont respectivement l'entrée et la sortie du quantificateur.

Le problème qui se pose est celui de l'existence de cette limite. si on considère les x_i comme étant des réalisations d'un procesus aléatoire x, et si ce processus est stationnaire et ergodique [19,31] alors cette limite tendra vers E(d(x,Q(x))) ; E étant l'espérance mathématique ; donc :

$$D = E[d(X, Q(X)]$$

Cette espérance mathématique, quand elle existe peut être décomposée de la manière suivante :

$$D = \sum_{i=1}^{L} p(x \in C_i) E[d(X, y_i)/x \in C_i]$$

x est une réalisation de X et y_i un élément de l'ensemble de prototypes Y.

$P(x \in C_i)$ est la probabilité discrète pour que x soit dans la classe C_i .

On obtient finalement :

$$D = \sum_{i=1}^{L} p(x \in C_i) \int_{x \in C_i} d(x, y_i) p(x) dx \qquad (3.1)$$

La minimisation de D nécessite d'abord la minimisation de $\int_{x \in C_i} d(x, y_i) p(x) dx$.

5 QUANTIFICATION VECTORIELLE SPHERIQUE

Cette technique a la particularité d'utiliser directement le signal temporel (échantillonné). On découpe le signal en blocs de m points. chaque bloc sera considéré comme un vecteur X ayant m composantes :

$$X = (x1, x2, ..., x_m)'$$

' dénote l'opération de transposition

La quantification vectorielle sphérique [1] dont l'interprétation géométrique est assez facile à comprendre, consiste à quantifier séparément la norme G (ou gain) et le vecteur normalisé (ou orientation) du vecteur X, soit :

$G^2 = \sum_{i=1}^{m} x_i^2$ la norme et
$x = x/G$ l'orientation.

Pour cette méthode on suppose connu un ensemble de N_0 prototypes. tous ces prototypes ont une norme égale à 1 ($\|y_i\| = 1$) pour tout i=1, 2, ... , N_0 . On cherche à quantifier un vecteur normé x, ceci revient à trouver le prototype y_i qui soit le plus proche possible du vecteur x.

La quantification de l'orientation met en oeuvre un premier quantificateur Q_0 auquel est associé la table de décodage $S_0 = \{y_i/i = 1, 2, ..., N_0\}$. On dira que ce quantificateur opère à un débit $R_0 = log_2(N_0)$. Ce quantificateur fournit le vecteur y_i qui minimise la distance :

$$d(x, y_i) = \|x - y_i\|^2$$
$$= \|x\|^2 + \|y\|^2 - 2x'y_i$$
$$= 2 - 2x'y_i$$

Minimiser la distance entre x et y_i revient à maximiser le produit scalaire $x'y_i$ lequel produit s'interprète comme la projection de l'orientation x sur y_i (ou vice-versa), donc :

$$max_i \{x'y_i\} = max_i \{proj(x)/y_i\} = P \quad (3.2)$$

Ou encore cela revient à maximiser la projection du vecteur non normalisé X :

$$max_i \{X'y_i\} = max_i \{proj(X)/y_i\} = PG \quad (3.3)$$

La figure 3.7 résume le mécanisme de la quantification vectorielle sphérique. Py_i *et* PGy_i fournissent respectivement une meilleure approximation des vecteurs x *et* X.

Il faut ensuite quantifier la norme G (quantificateur Q_G). En examinant la formule (3.3) on remarque qu'il est plus intéressant de quantifier PG car on n'aura pas à calculer explicitement la norme G (figure 3.8).

Finalement on obtient :

$$Y_k = Q(x) = Q_0(x)Q_G(PG)$$

Il y a plusieurs avantages à quantifier séparément norme et orientation :

1. Traiter la norme à part réduit de un la dimension du problème et permet donc d'utiliser un ensemble de prototypes de taille réduite. On gagne une dimension pour chaque prototype et de plus on utilise soit très peu de représentants soit un codage quelconque pour le gain.
2. Les avantages les plus importants viennent du fait que la norme présente certaines particularités distinctives que l'on peut exploiter. Le gain varie de façon lente d'un vecteur à l'autre. On peut exploiter facilement cette forme de redondance par un codage de type prédictif.

6 GENERATION DES PROTOTYPES

Il existe différents algorithmes d'apprentissage pour la génération des prototypes. Nous allons en présenter trois, le premier pour sa simplicité de conception, le deuxième pour sa

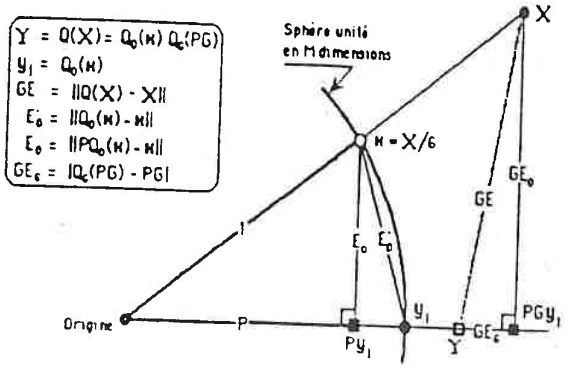


Figure 3.7: situation typique de quantification vectorielle sphérique. la sphère de rayon unité est représentée par un arc de cercle. Le vecteur y_i retenu est celui qui conduit à la plus grande projection P [1].

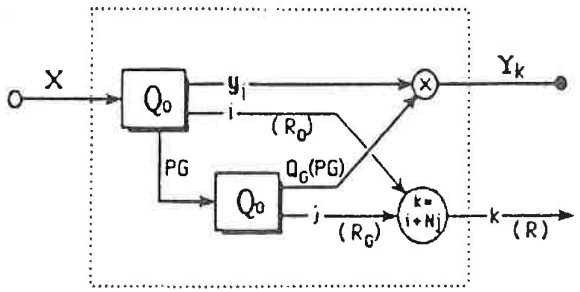


Figure 3.8: diagramme de la quantification vectorielle sphérique. Le quantificateur global symbolisé par le rectangle en pointillés se compose de deux quantificateurs l'un opérant sur l'orientation et l'autre sur PG [1]

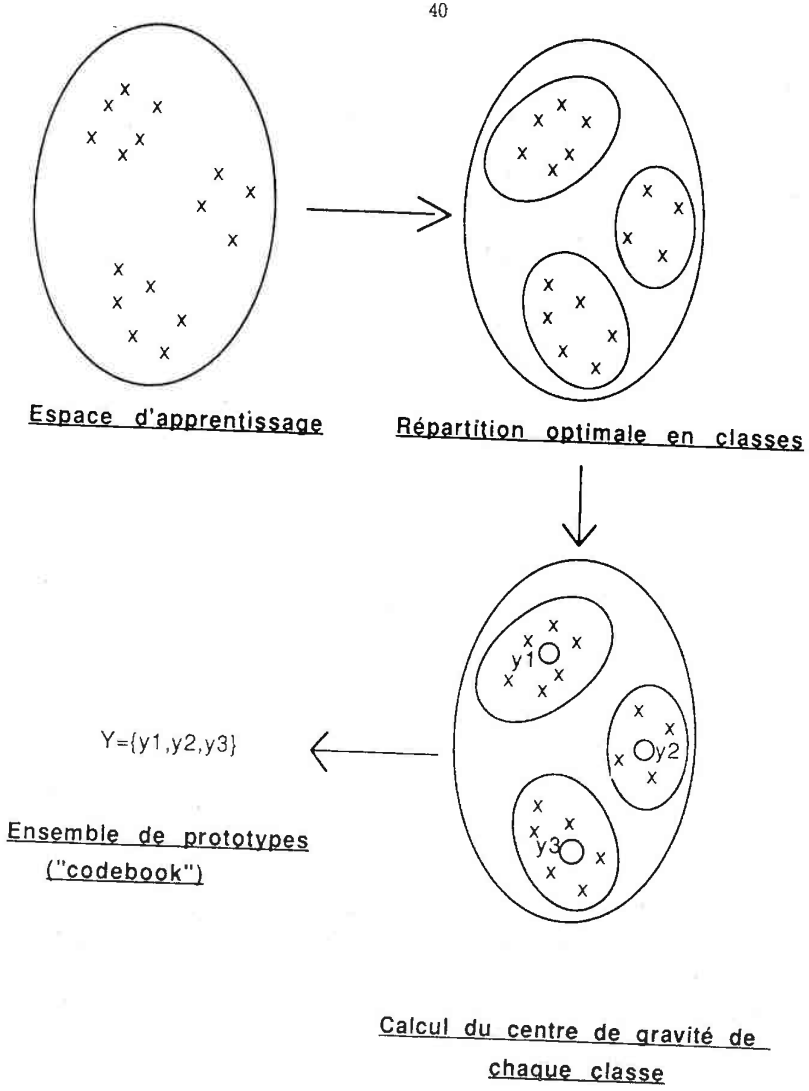


Figure 3.9: principe des algorithmes d'apprentissage

recherche exhaustive et le troisième pour sa hiérarchie. Ces algorithmes, sauf le premier, obéissent au principe général de la figure 3.9.

6.1 Algorithme "A Seuil"

Cet algorithme commence par considérer le premier vecteur du corpus d'apprentissage comme le représentant d'une classe dont il est le seul élément. Si la distance entre le deuxième vecteur et le représentant de la première classe est supérieure à un seuil donné, le deuxième vecteur sera le représentant et le seul élément d'une deuxième classe, sinon il sera le deuxième élément de la première classe dont on réactualise le représentant. On continue ce processus (à chaque arrivée, création d'une nouvelle classe ou réactualisation d'une classe déjà existante) jusqu'à épuisement de tout le corpus d'apprentissage ou jusqu'à ce que le nombre de prototypes voulus soit atteint.

L'avantage incontestable de cet algorithme est la rapidité (chaque entrée nécessite au plus c comparaisons; c étant le nombre de classes déjà créées), l'ensemble de prototypes est obtenu en un seul passage. Parmi les inconvénients on peut citer :

1. Etant donné un corpus d'apprentissage on ne peut pas contrôler avec précision le nombre de classes,
2. Du fait de sa réactualisation, le représentant d'une classe varie, d'où risque d'avoir des classes allongées (effet de chaîne),
3. Cet algorithme est sensible à l'ordre d'arrivée des vecteurs constituant le corpus d'apprentissage,
4. Il n'y a pas un fondement théorique qui valide le choix de cet algorithme. Cependant cet algorithme est utilisé et il semble que ces inconvénients ne sont en partie que théoriques [35].

6.2 algorithme de "LBG" (Linde,Buzo et Gray)

Rappel

Un quantificateur optimal est un quantificateur qui minimise la relation (3.1) ce qui revient à [15,31] :

1. Choisir une distorsion minimale entre un vecteur x et un prototype y_i. Ceci est réalisé en utilisant la règle du plus proche voisin (ppv) :

$$Q(x) = y_i \quad \text{si} \quad d(x, y_i) \leq d(x, y_j) \quad \text{pout tout j tel que} \quad j \neq i$$

où y_i et y_j appartiennent à l'ensemble de prototypes.

2. Chaque prototype y_i doit réaliser une distorsion moyenne minimale pour tous les éléments de la classe C_i.
   y_i est donc le vecteur qui minimise :

$$E[d(x,y)/x \in C_i] = \int_{x \in C_i} d(x,y)p(x)dx$$

Cette espérance mathématique est minimisée si on choisit pour y_i le centre de gravité (ou centroïde) de la classe C_i.

La recherche du centre de gravité d'une classe n'est pas un problème simple, la solution dépend de la distorsion utilisée.

L'algorithme des "K-means" optimise ces deux conditions, mais la recherche est itérative et demande beaucoup de temps.

Algorithme des "K-means"

Etant donné k prototypes et un corpus d'apprentissage $\{x_i; 1 \leq i \leq n\}$, cet algorithme fournit en sortie k prototypes réalisant une erreur de quantification plus petite que celle des prototypes de départ (figure 3.10). Le principe de cet algorithme consiste à coder tout le corpus d'apprentissage par l'ensemble de prototypes de départ. Ce codage fait apparaître des agglomérations (ou nuages) de points autour de chaque prototype, ensuite une erreur de quantification globale est évaluée. Si cette erreur est inférieure à un seuil donné on arrête le processus ; sinon, on remplace chaque classe par son centre de gravité et on réitère le processus jusqu'à ce que l'erreur de quantification soit faible.

Cet algorithme peut être résumé ainsi :

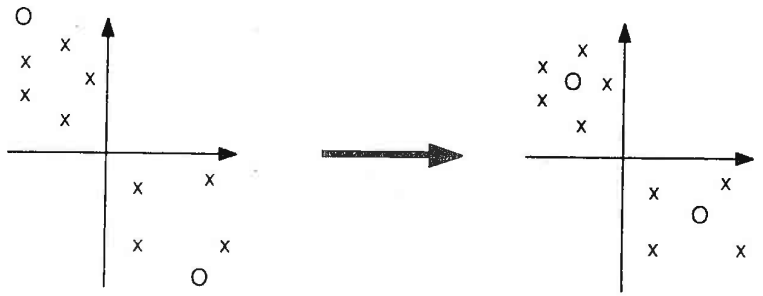


Figure 3.10: K-means : on obtient le même nombre de prototypes qu'au départ mais avec une faible erreur de quantification

Initialisation
$Y = \{y_0, y_1, ..., y_k\}$ ensemble de prototypes de départ et
$X = \{x_0, x_1, ..., x_n\}$ corpus d'apprentissage

Tant que l'erreur de quantification est supérieure à un seuil S faire

Codage du corpus d'apprentissage par $\{y_i\ ;\ 1 \leq i \leq k\}$ en utilisant la règle des plus proches voisins.
Le corpus est donc partitionné en classes $\{C_i\ ;\ 1 \leq i \leq k\}$

Calcul de l'erreur de quantification :

$$D = \frac{1}{n} \sum_{i=1}^{n} d(x_i, y_j)$$

y_j est le représentant de la classe C_j à laquelle est associé x_i

Réactualisation de l'ensemble de prototypes. Chaque prototype est remplacé par le centroïde de la classe dont il est le représentant.

Fin

Cet algorithme dépend à la fois du premier ensemble de prototypes et du seuil (test d'arrêt). Il existe différentes méthodes pour initialiser cet algorithme :

1. Prendre les k premiers vecteurs du corpus d'apprentissage pour le premier ensemble de prototypes, mais il se peut que ces premiers vecteurs soient corrélés entre eux.
2. Prendre un ensemble quelconque de vecteurs, par exemple la base de l'espace vectoriel.
3. Construction progressive des prototypes : soit on part du centre de gravité de tout le corpus d'apprentissage [28], soit on part, si les vecteurs sont paramétrés par les coefficients de réflexion k_i, des deux vecteurs $k = [+/-0.5, 0, 0, ..., 0]$ [25]. Par des techniques de perturbation "splitting" chaque prototype donne naissance à deux prototypes et ainsi de suite jusqu'à ce qu'on obtienne l'ensemble de prototypes voulu.

Algorithme de "LBG" [28]

Cet algorithme impose que le nombre k de prototypes soit une puissance de deux. Il reprend l'algorithme des "K-means" en partant d'un ensemble de prototypes ayant un seul élément. Cet élément est le centre de gravité de tout le corpus d'apprentissage. On construit progressivement l'ensemble de prototypes.

Initialisation
Calcul du centre de gravité de tout le corpus d'apprentissage (y_0).
L'ensemble de prototypes Y_0 contient un seul élément :

$$Y_0 = \{y_{01}\}$$

Tant que le nombre de prototypes voulu n'est pas atteint faire

% à l'étape m on a : $Y_m = \{y_{m1}, ..., y_{m2^m}\}$ %
Eclatement ou perturbation ("Splitting") de tous les prototypes.
L'ensemble de prototypes devient :

$$Y'_m = \{y_{m1} - \varepsilon, y_{m1} + \varepsilon, ...\}$$

ε est un vecteur de faible norme

"K-means"
Les nouveaux prototypes qu'on obtient après perturbation ne sont pas forcément ceux qui introduisent le moins d'erreurs de quantification. On affine cette hypothèse par application de l'algorithme des "K-means" sur les 2^m prototypes.

Fin

Cet algorithme prend beaucoup de temps (le codage de chaque vecteur nécessite 2^p comparaisons; p étant le nombre de prototypes), c'est en fait la partie "K-means" qui demande énormément de comparaisons car à chaque étape il faut recoder tout le corpus.

On remarque aussi que les prototypes obtenus sont des centres de gravité, ils n'ont donc pas d'existence physique. Ce sont des vecteurs (comme on le verra plus loin) obtenus par "moyennage" des vecteurs qui constituent la classe. Pour éviter ces prototypes artificiels on peut garder par exemple pour chaque classe le vecteur réel le plus proche du centre de gravité.

L'exploitation des prototypes qu'on obtient est coûteuse car il n'y a aucune hiérarchie entre les prototypes. Il faut faire 2^p comparaisons pour trouver le plus proche voisin dans un ensemble de p prototypes. Au delà de p=10 l'ensemble de prototypes devient inexploitable surtout pour les applications temps réel.

Si on n'utilise pas un grand corpus d'apprentissage il risque d'y avoir des classes vides, on peut par exemple convenir de supprimer le représentant de ces classes.

Pratiquement, si on choisit un seuil raisonnable la convergence est obtenue au bout d'une dizaine d'itérations pour chaque nouvel ensemble de prototypes (figure 3.16), mais le nombre de comparaisons croît de façon exponentielle.

Enfin une amélioration possible de cet algorithme consiste à perturber d'avantage les prototypes des classes ayant une forte agglomération ou les classes réalisant une grande distorsion moyenne.

6.3 Algorithme "Arbre binaire"

Cet algorithme est une variante hiérarchisée de l'algorithme de "LBG". Par rapport à ce dernier on gagne en rapidité mais on perd en précision.

Le début de cet algorithme peut se résumer ainsi :

1. $Y0 = \{y\}$ centre de gravité du corpus d'apprentissage.
2. Après perturbation on obtient :
   $Y'1 = \{y'_0, y'_1\}$
3. Répartition du corpus par rapport à y'_0 et y'_1, on obtient deux corpus (corpus1 et corpus2)
4. Calcul de y_0 centre de gravité du corpus1 et y_1 centre de gravité du corpus2. L'ensemble de prototypes est donc :
   $Y1 = \{y_0, y_1\}$
5. On recommence les opérations 2, 3 et 4 jusqu'à ce qu'on obtienne le nombre de prototypes voulus (figure 3.11).

L'exploitation de l'ensemble de prototypes qu'on obtient est assez rapide (2p comparaisons au lieu de 2^p) mais rien ne permet d'affirmer qu'on n'a pas laissé le bon prototype dans l'autre branche de l'arbre.

6.4 Conclusion

Tous ces algorithmes de classification automatique acceptent toutes les formes de paramétrisations et de distorsions. On trouve d'autres variantes de ces algorithmes les uns hiérarchisant les prototypes, les autres étudiant le problème d'allocation des bits [16,8,57,53] Le problème qui se pose, comme on le verra plus loin, est le calcul du centre de gravité.

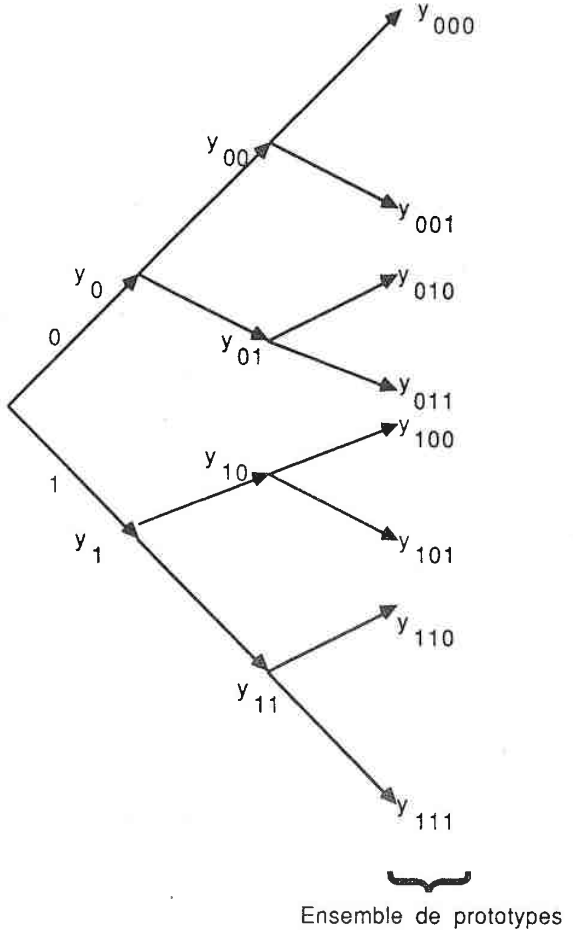


Figure 3.11: $Y = \{y_{000}, y_{001}, \ldots, y_{111}\}$ est l'ensemble de prototypes obtenus par recherche binaire.

7 Distorsions

Pour qu'elle soit utilisable une distorsion doit se prêter aux études théoriques et être facilement calculable. Ce qu'on attend le plus d'une distorsion est que tout changement de sa valeur corresponde concrètement à un changement de qualité de parole.

La plupart des distorsions utilisées proviennent de la forme générale suivante :

$$d(x,y) = (x-y)'M(x-y)$$

$$= \sum_i \sum_j (x_i - y_i) m_{ij} (x_j - y_j)$$

où x = (x1, ... , x_p)' et y = (y1, ... , y_p)' et
M est une matrice définie positive.

La matrice M a pour effet d'introduire une pondération et donc de rendre la contribution de certains éléments plus importante que d'autres.

Parmi ces distorsions on peut citer :

1. Le carré de la distance euclidienne

$$d(x,y) = (x-y)'I(x-y) = \sum_{k=1}^{p} (x_k - y_k)^2$$

Bien qu'elle ne soit pas subjectivement signifiante, c'est la distorsion la plus utilisée car elle se calcule facilement et elle est mathématiquement maniable.

La forme générale de cette distorsion est :

$$d(x,y) = \sum_{k=1}^{p} |x_k - y_k|^q$$

2. La distance de Mahalanobis

$$D(x,y) = (x-y)'B^{-1}(x-y)$$

où B est la matrice des covariances de la variable aléatoire X :

$$B = E[(X-m)(X-m)'] \qquad avec \quad m = E[X]$$

3. La distorsion d'Itakura-Saito

Cette distorsion est spécifique à la paramétrisation par LPC. Elle minimise la différence entre le spectre d'une fenêtre d'analyse et le spectre du modèle autorégressif. Cette distorsion admet plusieurs versions, les unes utilisées pour leur maniabilité mathématique, les autres pour leur facilité de calcul.
Cette distorsion à la forme suivante [6,41] :

$$D_{IS} = \frac{1}{2\pi} \int_0^{2\pi} [\frac{S_{in}(\omega)}{S(\omega)} + \log \frac{S(\omega)}{S_{in}(\omega)} - 1] d\omega$$

où $S_{in}(\omega)$ est le spectre de la fenêtre d'analyse (ou signal d'entrée) et $S(\omega)$ est le spectre d'un modèle (LPC), modèle de référence par exemple. Le spectre d'un modèle LPC est donné par :

$$S(\omega) = \frac{\sigma^2}{|1 + a_1 e^{-j\omega} + ... + a_p e^{-jp\omega}|^2}$$

Cette même distorsion peut être utilisée pour mesurer la différence entre les spectres de deux modèles (LPC) $S_R(\omega)$ *et* $S_T(\omega)$. Elle peut être calculée par :

$$D_{IS} = \frac{a_R' R_T a_R}{\sigma_T^2} + \log \frac{\sigma_T^2}{\sigma_R^2} - 1$$

où $a_R = (1, a_1, ..., a_p)'$
R_T la matrice des autocorrélations de $S_T(\omega)$
σ_T *et* σ_R les gains respectifs de $S_T(\omega)$ *et* $S_R(\omega)$.

L'inconvénient de cette distortion est qu'elle est sensible au gain du modèle. La distorsion entre une forme et la même forme amplifiée est non nulle :

$$\log \frac{\sigma_T^2}{\sigma_R^2} \neq 0$$

Cet inconvénient peut être la source de beaucoup d'erreurs dues aux conditions d'enregistrement de la parole.

Pour remédier à cela on utilise une autre forme de cette distorsion où l'on élimine l'effet du gain. C'est la distorsion connue sous le nom de la distorsion d'Itakura-Saito à gain normalisé dont l'expression mathématique est :

$$D_{GN} = D_{IS}(\frac{S_R(\omega)}{\sigma_R^2}, \frac{S_T(w)}{\sigma_T^2}) = \frac{a_R' R a_R}{\sigma_T^2} - 1$$

Il existe encore une autre forme appelée distorsion d'Itakura-Saito à gain optimisé :

$$D_{GO} = \log(1 + D_{GN}) = \log \frac{a_R' R a_R}{\sigma_T^2}$$

Pour ces deux dernières formes de la distorsion d'Itakura-Saito, on peut regretter le fait qu'on perde toute information pouvant provenir du gain. Il existe d'autres formes de cette distorsion combinant d'une certaine manière le gain [47,41,39].

La distorsion D_{GN} est la distorsion la plus utilisée car il n'y a pas de calcul de logarithme. Nous utiliserons pour la génération de l'ensemble de prototypes la distorsion D_{GN} qui admet la version calculable suivante :

$$D_{GN}(X, A) = \frac{r_x(0)}{\sigma_x^2} r_a(0) + 2 \sum_{i=1}^{p} \frac{r_x(i)}{\sigma_x^2} r_a(i) - 1$$

où $r_x(i)$ sont les coefficients d'autocorrélation du vecteur X,
$r_a(i)$ les coefficients d'autocorrélation du modèle A et
σ_x le gain du modèle LPC du vecteur X.

Les expressions mathématiques de $r_x(i)$ *et* $r_a(i)$ sont :

$$r_x(i) = \sum_{k=0}^{N-i-1} x(k)x(k+i) \qquad i = 0, ..., p$$

$$r_a(i) = \sum_{k=0}^{p-i} a_k a_{k+i} \qquad i = 0, ..., p$$

où x(k) est un point du signal numérique et
N la longueur de la fenêtre d'analyse.

Etant donnée la forme de cette distorsion il convient de paramétrer le corpus d'apprentissage par ses coefficients d'autocorrélation normalisés par le gain et les prototypes par les coefficients d'autocorrélation du modèle.

7.1 Calcul du centre de gravité

Nous avons vu que pour la génération des prototypes il fallait à chaque fois remplacer les anciens prototypes par le centre de gravité de la classe dont ils sont le représentant. Le calcul du centre de gravité dépend de la distorsion utilisée. Donnons l'expression mathématique pour deux distorsions :

1. Carré de la distance euclidienne

   Le centre de gravité est l'élément u qui minimise :

   $$\frac{1}{m} \sum_{i=1}^{m} (x_i - u)^2$$

   En annulant la dérivée par rapport à u on obtient :

   $$u = \frac{1}{m} \sum_{i=1}^{m} x_i$$

   Le vecteur centre de gravité est obtenu tout simplement par moyennage de tous les vecteurs qui forment la classe.

2. Distorsion d'Itakura-Saito à gain normalisé

   Cette fois, pour des raisons de commodité du calcul, on utilise pour la distorsion d'Itakura-Saito à gain normalisé la forme quadratique [25] :

   $$d_{GN}(S_R, S_T) = \frac{(a_R - a_T)' R_T (a_R - a_T)}{\sigma_T^2}$$

   où R_T est la matrice des autocorrélations du modèle S_T . Cette matrice est symétrique définie positive.

Le vecteur centre de gravité est le vecteur qui minimise la fonction :

$$f(u) = \sum_{T=1}^{m} (u - a_T)' \frac{R_T}{\sigma_T^2} (u - a_T)$$

On a :

$$f(u + du) = f(u) + 2 \sum (u - a_T)' \frac{R_T}{\sigma_T^2} du + \sum du' \frac{R_T}{\sigma_T^2} du$$

Minimiser cette quantité revient à annuler la dérivée première par rapport à u, soit :

$$\sum (u - a_T)' \frac{R_T}{\sigma_T^2} = 0$$

qu'on peut écrire encore :

$$\sum \frac{R_T}{\sigma_T} u = \sum \frac{R_T}{\sigma_T} a_T \tag{3.4}$$

et donc :

$$u = (\sum \frac{R_T}{\sigma_T^2})^{-1} \sum \frac{R_T}{\sigma_T^2} a_T \tag{3.5}$$

Le calcul de u à partir de la formule (3.5) est compliqué et demande beaucoup de temps. Il suffit de remarquer que la relation (3.4) est justement le système LPC (2.8) du chapitre 2. Les coefficients LPC du vecteur centre de gravité sont obtenus, malgré la complexité de la distorsion, par simple moyennage des coefficients d'autocorrélation de tous les vecteurs qui forment la classe, et par application de la procédure Levinson/Durbin.

8 Choix des paramètres à quantifier

Parmi les étapes que comportent les algorithmes d'apprentissage automatique on trouve l'opération de perturbation ("splitting") des vecteurs. Cette opération demande quelques précautions car les vecteurs qu'on perturbe vérifient souvent une ou plusieurs propriétés et il faut que les vecteurs après perturbation continuent à vérifier au moins la propriété principale. Par exemple on ne peut pas perturber les coefficients LPC car la propriété de stabilité risque de ne plus être vérifiée. Pour mieux cerner ce problème on va étudier les paramètres qui résistent mieux aux perturbations et donc à la quantification.

Rapppelons qu'un modèle autorégressif utilisé dans un système de prédiction linéaire a une fonction de transfert de la forme :

$$H(z) = \frac{G}{A(z)} = \sum_{n=0}^{\infty} h_n z^{-n}$$

où A(z) est le filtre inverse donné par :

$$A(z) = 1 + \sum_{n=1}^{p} a_n z^{-n}$$

et G le gain

On va donner une liste de paramètres qui caractérisent uniquement le filtre H(z) :

1. Les coefficients LPC $(1, a_1, ..., a_p)$
2. La réponse impulsionnelle du filtre h_n
3. Les coefficients d'autocorrélation de $\{a_n/G\}$:

$$b_i = \frac{1}{G^2} \sum_{j=0}^{p-|j|} a_j a_{j+|i|} \qquad a_0 = 1, \quad 0 \leq i \leq p$$

4. Les coefficients d'autocorrélation de $\{h_n\}$

$$r_i = \sum_{j=0}^{\infty} h_j h_{j+|i|} \qquad 0 \leq i \leq p$$

5. Les coefficients cepstraux de $\{A(z)/G\}$:

$$P_i = b_0 + 2\sum_{j=1}^{p} b_j \cos\frac{2\pi ij}{2p+1} \qquad 0 \le i \le p$$

les b_i sont les mêmes que dans 3.

6. Les coefficients cepstraux de A(z) :

$$c_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log A(e^{j\omega}) e^{jn\omega} d\omega$$

Ces coefficients peuvent être calculé à partir des coefficients LPC par les relations suivantes :

$$c_1 = a_1$$

$$c_n = a_n - \sum_{m=1}^{n-1} \frac{m}{n} c_m a_{n-m} \qquad 2 \le n \le p$$

7. Les pôles de H(z) (ou zéros de A(z))
8. Les coefficients de réflexion $\{k_i \quad 1 \le i \le p\}$
9. Les rapports d'aire :

$$A_i = A_{i+1}\frac{1+k_i}{1-k_i} \qquad A_{p+1} = 1 \qquad 1 \le i \le p$$

Les coefficients du filtre LPC peuvent être retrouvés à partir de tous ces ensembles de paramètres.

Dans un but de quantification il est souhaitable qu'un ensemble de paramètres possède les deux propriétés suivantes :

1. Le filtre reste stable après quantification.
2. Les coefficients sont dans un ordre naturel.

La stabilité du filtre signifie que les pôles de H(z) doivent rester à l'intérieur du cercle unité.

Les coefficients sont dans un ordre naturel si leur fonction de transfert change quand on permute deux coefficients.

Exemple :
La fonction de transfert d'un modèle LPC peut s'écrire :

$$H(z) = \frac{1}{1+\sum a_i z^{-i}} = \frac{1}{\prod(1-p_i)}$$

où a_i sont les coefficients LPC et p_i les pôles de H(z)

Les coefficients LPC sont dans un ordre naturel car la fonction H(z) change quand on permute a_i et a_j . Par contre les pôles ne sont pas dans un ordre naturel car H(z) reste la même quand on inter-change deux pôles p_i et p_j .

Quand un ensemble de paramètres possède un ordre naturel il est possible d'étudier statistiquement le comportement individuel de chaque paramètre et d'utiliser ainsi un codage plus économique et mieux adapté pour chaque paramètre.

Les coefficients a_n et h_n peuvent causer l'instabilité du filtre après quantification. Les coefficients b_i et r_i forment une matrice définie positive, or cette propriété n'est pas assurée après quantification. Les coefficients h_n et r_n peuvent être utilisés mais avec un minimum de quantification et donc avec un fort débit de transmission.

Parmi tous ces paramètres seuls les coefficients de réflexion et les pôles assurent la stabilité après quantification mais seuls les coefficients de réflexion possèdent un ordre naturel. Cette dernière propriété étant surtout nécessaire pour la transmission à bas débit. On peut cependant ordonner les pôles à l'aide des formants mais cette solution demande un calcul complexe et coûteux.

On peut se demander quelle serait la quantification optimale donnant la meilleure qualité d'écoute quand on synthétise la parole à partir des coefficients de réflexion quantifiés. La quantification optimale des coefficients de réflexion a été largement étudié dans [56,18,17,32]. Pour étudier mathématiquement cette quantification optimale il faut définir un critère de choix ; comme notre oreille est sensible au côté spectral ce critère doit être fondé sur une analyse spectrale.

Viswanathan et Makhoul [56] ont défini la déviation spectrale dûe à la perturbation du k-ième coefficient de réflexion ($k_i \quad 1 \le i \le p$) par :

$$\Delta S = \frac{1}{2\pi}\int_{-\pi}^{\pi} |\log P(k_i,\omega) - \log P(k_i + \Delta k_i,\omega)| d\omega$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} \log\frac{P(k_i,\omega)}{P(k_i+\Delta k_i,\omega)} d\omega$$

où $P(.,\omega) = |H(e^{j\omega})|^2$

et la sensibilité spectrale par :

$$\frac{\partial S}{\partial k_i} = \lim_{\Delta k_i \to 0} |\frac{\Delta S}{\Delta k_i}|$$

$$= \lim_{\Delta k_i \to 0} |\frac{1}{\Delta k_i}[\frac{1}{2\pi}\int_{-\pi}^{\pi} |\log \frac{P(k_i,\omega)}{P(k_i+\Delta k_i,\omega)}|d\omega]|$$

La figure 3.12 montre 12 courbes de déviations spectrales ($10\log_{10}(\frac{\partial S}{\partial k_i})$) tracées par Viswanathan et Makhoul. Ces courbes sont obtenues à partir de modèles LPC d'ordre 12 estimés sur des fenêtres de 20 ms ; la fréquence d'échantillonnage du signal est de 10 Khz. Chaque courbe représente la sensibilité spectrale d'un coefficient k_i quand sa valeur varie entre -1 et 1 et les autres coefficients maintenus constants. On remarque que :

1. Chacune des 12 courbes correspondant aux 12 coefficients de réflexion a la même allure, indépendamment du numéro du coefficient et des valeurs des autres coefficients.

2. Chaque courbe a une forme en U et est symétrique par rapport à $k_i = 0$. De plus la sensibilité est beaucoup plus grande lorsque $|k_i|$ est proche de 1 que de zéro. Lors du codage, une erreur, aussi faible soit elle, de quantification de l'un des coefficients de réflexion proche de l'unité peut introduire des variations spectrales importantes.

La figure 3.12 montre la sensibilité spectrale pour chaque coefficient de réflexion, or il est intéressant de connaître en moyenne le comportement de l'ensemble des paramètres et sur plusieurs fenêtres successives.

La figure 3.13 montre la sensibilité moyenne de tous les coefficients de réflexion et sur N fenêtres, cette sensibilité moyenne est donnée par :

$$\frac{\overline{\partial S}}{\partial k} = \frac{1}{pN}\sum_{j=1}^{N}\sum_{i=1}^{p}\frac{\partial S}{\partial k_{ij}}$$

k_{ij} est le i-ème coefficient de réflexion (k_i) pour la fenêtre j.

On remarque que la courbe ($10\log\frac{1}{1-k^2}$) constitue une bonne approximation de la courbe de sensibilité spectrale moyenne.

En conclusion une quantification linéaire des coefficients de réflexion n'est certainement pas une solution satisfaisante surtout pour les valeurs qui approchent 1. Il faut adopter une quantification non linéaire qui autorise plus d'espacement autour de 1 que

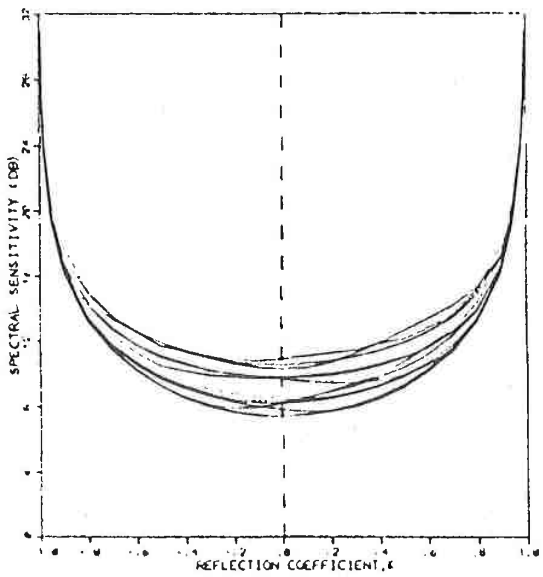


Figure 3.12: Courbes de déviation spectrale pour les coefficients de réflexion d'un modèle à 12 pôles estimés sur une fenêtre de 20 ms.

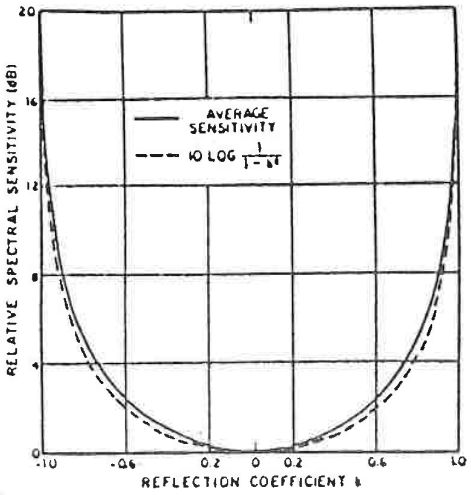


Figure 3.13: Courbe de la sensibilité spectrale moyenne pour les coefficients de réflexion et une fonction qui en constitue une approximation

de 0. Cette quantification non linéaire revient à quantifier linéairement des paramètres obtenus par transformation non linéaire des coefficients de réflexion.

Soient g_i les coefficients obtenus par transformation non linéaire des coefficients k_i :

$g_i = f(k_i)$

Pour éviter les problèmes posés par les coefficients de réflexion k_i aux deux extrémités, on impose que les nouveaux paramètres aient une sensibilité spectrale plate c'est à dire :

$$\frac{\partial S}{\partial g_i} = L = constante \tag{3.6}$$

On calcule de la même manière la sensibilité spectrale :

$$\frac{\partial S}{\partial g_i} = \frac{\partial S}{\partial k_i}\frac{\partial k_i}{\partial g_i} = \frac{\frac{\partial S}{\partial k_i}}{\frac{df(k_i)}{dk_i}} \tag{3.7}$$

En remplaçant (3.6) dans(3.7) on obtient :

$$\frac{df(k_i)}{dk_i} = \frac{1}{L}\frac{\partial S}{\partial k_i} \tag{3.8}$$

Il suffit d'intégrer l'équation (3.8) pour obtenir la transformation optimale des coefficients k_i . La relation (3.8) doit être vérifiée pour tous les coefficients k_i donc vérifiée par la fonction de la sensibilité spectrale moyenne de la figure 3.13.

Une approximation de cette solution revient à intégrer la fonction tracée en pointillés (figure 3.13) au lieu de la fonction de la sensibilité spectrale moyenne, donc :

$$\frac{df(k_i)}{dk_i} = \frac{1}{L(1-k_i^2)}$$

La solution est :

$$f(k_i) = \frac{1}{2L}\log\frac{1+k_i}{1-k_i}$$

Comme L est une constante arbitraire, avec $L = \frac{1}{2}$ on obtient :

$$f(k_i) = \log \frac{1+k_i}{1-k_i}$$

Ces coefficients sont les rapports d'aire logarithmiques associés aux k_i ou LAR (Log Area Ratio). La figure 3.14 montre que la sensibilité spectrale de ces nouveaux paramètres est presque plate et qu'on n'a plus le problème des extrémités.

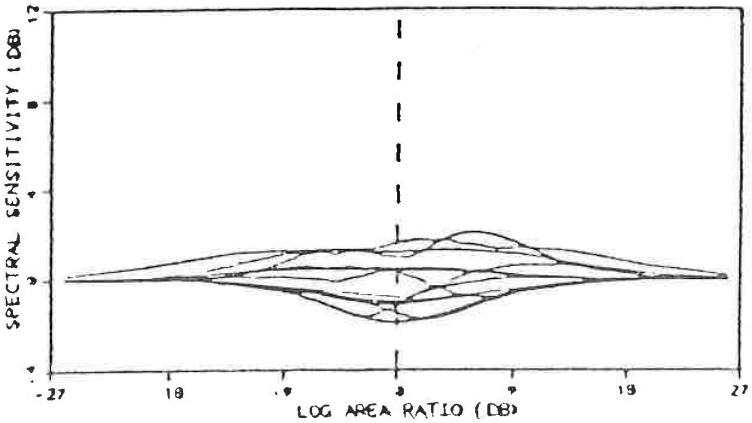


Figure 3.14: Courbes de sensibilité spectrale en utilisant les coefficients L.A.R

Les coefficients LAR sont les coefficients qui résistent mieux à la quantification. La figure 3.15 montre l'évolution des coefficients LAR en fonction des coefficients de réflexion (k_i). On remarque que pour les valeurs de k_i inférieurs à 0.7 une quantification linéaire est suffisante. En pratique les coefficients de réflexion k_i pour $i > 3$ ont généralement une amplitude < 0.7.

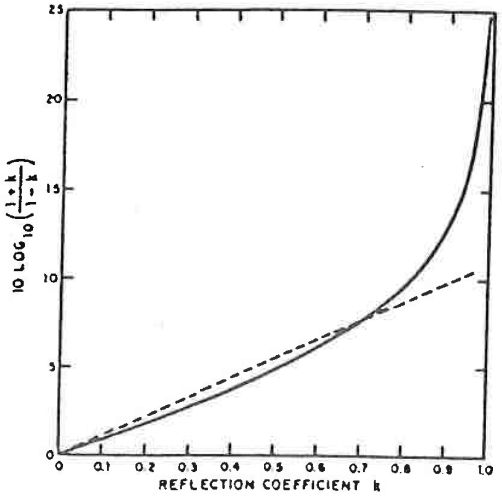


Figure 3.15: évolution des coefficients LAR en fonction de k_i. la courbe et la droite ont une intersection en $k_i = 0.7$.

9 Algorithme proposé

Nous avons vu dans ce chapitre que la distorsion d'Itakura-Saito à gain normalisé est un distorsion qui a beaucoup d'avantages. en particulier du fait qu'elle est fondée sur la mi imisation des spectres et donc tient compte de la perception humaine. On a aussi prou que les coefficients LAR obtenus par transformation non linéaire des coefficients de réflexio sont les paramètres qui supportent mieux les effets de la quantification.

Nous proposons un algorithme de quantification qui est une version de l'algorithme "LBG utilisant la distorsion d'Itakura-Saito à gain normalisé. Pour éviter le calcul du logarith et compte tenu de la figure 3.15 on a utilisé les coefficients de réflexion. Pour ne pas av de déviation spectrale importante au voisinage de 1 nous avons décidé de ne pas perturb les coefficients de réflexion k_1 *et* k_2 (on a remarqué expérimentalement qu'ils étaient seuls à approcher 1 et -1).

L'algorithme peut se résumer ainsi :

Initialisation
Calcul du centre de gravité (y_{01}) de tout le corpus d'apprentissage.
L'ensemble de prototypes initial Y_0 ne contient qu'un seul élément :
$Y_0 = \{ y_{01} \}$

Tant que le nombre de prototypes voulu n'est pas atteint faire

Perturbation
A l'étape m on dispose de l'ensemble de prototypes suivant :

$$Y_m = \{y_{m1}, y_{m2}, ... y_{m2^m}\}$$

Pour chaque prototype y_{mi} on transforme ses coefficients LPC (a_i) en coefficients de réflexion (k_i)
Pour chaque prototype on perturbe ses coefficients k_i $(3 \leq i \leq p)$
Pour chaque prototype, après perturbation. on transforme ses coefficients (k_i) en coefficients LPC (a_i)

"K-means" sur Y_m

Fin

La perturbation qu'on a utilisé pour les coefficients de réflexion est la suivante :

$k_{gi} = 0.999 * k_i$ perturbation gauche et
$k_{di} = 1.001 * k_i$ perturbation droite pour tout i tel que

$$3 \leq i \leq p$$

$(k_1, k_2, k_{g3}, ..., k_{gp})$ *et* $(k_1, k_2, k_{d3}, ..., k_{dp})$ sont les deux vecteurs obtenus par perturbation de $(k_1, k_2, ..., k_p)$.

L'étape de codage du corpus utilise les coefficients LPC (a_i) et l'étape de perturbation utilise les coefficients de réflexion (k_i). On a besoin de deux procédures qui permettent de passer des coefficients a_i aux coefficients k_i et vice-verca.

Cet algorithme malgré sa complexité ne donne qu'une solution sous optimale. En effet, les prototypes obtenus varient en fonction de la perturbation choisie et du seuil utilisé par l'algorithme "K-means".
La figure 3.16 montre une exécution de cette algorithme. A chaque étape on donne le nombre d'éléments constituant chaque classe ainsi que le nombre d'itérations pour obtenir la convergence de l'algorithme "K-means".

10 Conclusion

Dans ce chapitre nous avons mis en évidence les divers aspects de la quantification scalaire et vectorielle. Nous avons proposé une version de l'algorithme de "LBG" qui tient compte des problèmes de stabilité après quantification. Nous avons aussi présenté une liste de paramètres qui gardent le même spectre malgré les déformations dues à la quantification.

Dans le chapitre suivant nous traiterons les problèmes de transmission, en commençant par préciser l'information à transmettre, c'est à dire une information non redondante mais suffisante pour retrouver le signal de départ avec un minimum de confort d'écoute.

```
 2      distorsion    1.232783    iteration    1
 2      distorsion    1.220522    iteration    2
 2      distorsion    1.220295    iteration    3
  2229 2270

 4      distorsion    1.220245    iteration    1
 4      distorsion    1.213309    iteration    2
 4      distorsion    1.208725    iteration    3
 4      distorsion    1.207035    iteration    4
 4      distorsion    1.206288    iteration    5
 4      distorsion    1.205939    iteration    6
  1236 1063 1012 1188

 8      distorsion    1.205879    iteration    1
 8      distorsion    1.197374    iteration    2
 8      distorsion    1.194242    iteration    3
 8      distorsion    1.191665    iteration    4
 8      distorsion    1.190257    iteration    5
 8      distorsion    1.189620    iteration    6
 8      distorsion    1.189207    iteration    7
   602  615  361  557  576  541  586  661

16      distorsion    1.189143    iteration    1
16      distorsion    1.179820    iteration    2
16      distorsion    1.177172    iteration    3
16      distorsion    1.175433    iteration    4
16      distorsion    1.174388    iteration    5
16      distorsion    1.173696    iteration    6
16      distorsion    1.173181    iteration    7
  322  314  274  316  319  244  283  312  250  309  199  166  314  294  252  331

32      distorsion    1.173121    iteration    1
32      distorsion    1.164402    iteration    2
32      distorsion    1.161188    iteration    3
32      distorsion    1.158980    iteration    4
32      distorsion    1.157696    iteration    5
32      distorsion    1.156863    iteration    6
32      distorsion    1.156279    iteration    7
32      distorsion    1.155809    iteration    8
  172  170  144  121  142  157  155  165   88  112  120  139  145  136  134  145
  150  128  126  132  152  122  168  138  149  158  128  142  130  125  142  164

64      distorsion    1.155745    iteration    1
64      distorsion    1.147581    iteration    2
64      distorsion    1.144122    iteration    3
64      distorsion    1.142345    iteration    4
64      distorsion    1.141258    iteration    5
64      distorsion    1.140535    iteration    6
64      distorsion    1.140025    iteration    7
   75   84   66   70   67   70   41   65   73   60   59   64   89   68   62   70
   70   78   73   86   57   57   81   68   68   62   65   82   64   72   72   84
   61   77   61   70   91   63   69   71   63   78   50   67   54   57   67   52
   81   68   89   88   81   84   55   80   70   56   61   73   92   89   67   92
```

Figure 3.16: Exemple de génération de prototypes. Le signal utilisé représente le bruit de la salle d'enregistrement. Malgré la ressemblance des spectres l'algorithme réalise un partage assez équilibré.

Chapitre 4

SYNTHESE DE LA PAROLE

1 INTRODUCTION

On a vu dans le chapitre précédent quelques techniques de compression de données mais encore faut il savoir restituer l'information initiale.

La synthèse de la parole à partir d'un ensemble quelconque de paramètres (banc de filtres, LPC, formants, texte, ...) est un moyen auditif pour vérifier la validité de la paramétrisation choisie, et cela a constitué notre motivation essentielle pour cette partie de notre travail. La synthèse est aussi une nécessité pour les transmissions à faible débit.

Dans ce chapitre nous donnons une liste non exhaustive des techniques de restitution de la parole, en se limitant au cas de la restitution à partir de LPC ou de paramètres dérivés de LPC. On terminera en proposant un synthétiseur qui réalise un compromis entre le nombre de bits à transmettre et la qualité d'écoute.

Avant d'aborder le problème de la synthèse on va donner quelques rappels sur la théorie du filtrage numérique [46,26,40].

2 DEFINITIONS

2.1 SYSTEME

Un système de traitement numérique, ou système, agit sur un signal numérique d'entrée et produit un autre signal numérique à sa sortie. Le signal d'entrée est appelé signal d'excitation ou excitation. Le signal de sortie est appelé réponse du système.

Mathématiquement, un système est une transformation T qui agit sur un signal d'entrée x(n) et le transforme en un signal de sortie y(n). Cette opération est représentée formellement par :

$$y(n) = T[x(n)]$$

Exemple : La transformation T peut être la transformation de Fourier discrète TDF. Le système correspondant est appelé transformateur de Fourier.

2.2 IMPULSION UNITE

L'impulsion unité est définie par :

$$\delta(k) = \begin{cases} 1 & \text{pour k=0} \\ 0 & \text{pour } k \neq 0 \end{cases}$$

Tout signal numérique peut être exprimé comme une somme pondérée d'impulsions unité décalées :

$$x(k) = \sum_{l=-\infty}^{+\infty} x(l)\delta(k-l) \tag{4.1}$$

2.3 TRANSFORMATION EN Z

La transformée en z est une généralisation de la transformée de Fourier à laquelle elle peut s'identifier dans un cas particulier.

La transformée en z d'un signal x(n) est définie par :

$$X(z) = \sum_{n=-\infty}^{+\infty} x(n)z^{-n}$$

Où z est une variable complexe et X(z) une fonction complexe de la variable z.

La transformée en z est surtout utilisée dans les problèmes orientés vers l'analyse et la synthèse des systèmes de traitement (par exemple en filtrage numérique). Pour ces problèmes les limites des capacités de la transformée de Fourier sont vite atteintes.

La transformée en z permet, par exemple, de représenter un signal possèdant une infinité d' échantillons par un ensemble fini de nombres. Ces nombres, caractérisant complètement

le signal, permettent de le reconstituer entièrement.

La transformée en z d'une fenêtre d'analyse $\{x(n), \quad 0 \leq n \leq N-1\}$ est donnée par :

$$X(z) = \sum_{n=0}^{N-1} x(n)z^{-n}$$

En donnant à z la valeur $e^{j(\frac{2\pi}{N})k}$ on retrouve la définition de la transformée de Fourier discrète.

On appelle fonction de transfert d'un système la transformée en z de sa réponse impulsionnelle.

2.4 PRODUIT DE CONVOLUTION

Le produit de convolution de deux signaux numériques x(n) et y(n) est donnée par :

$$u(n) = \sum_{k=-\infty}^{+\infty} x(k)y(n-k)$$

Ce produit est souvent noté :

$$u(n) = x(n) \star y(n)$$

2.5 SYSTEME LINEAIRE

Un système linéaire est défini par le principe de superposition. Soient deux signaux numériques d'entrée $x_1(n)$ *et* $x_2(n)$, un système est linéaire si et seulement si pour toutes constantes a et b on a :

$$T[ax_1(n) + bx_2(n)] = aT[x_1(n)] + bT[x_2(n)]$$

$$= ay_1(n) + by_2(n)$$

où $y_1(n)$ *et* $y_2(n)$ sont les réponses aux excitations $x_1(n)$ *et* $x_2(n)$.

Un système linéaire est complètement caractérisé par sa réponse impulsionnelle. En effet, par application de l'opération T à la relation (4.1) on obtient :

$$y(k) = T[x(k)] = \sum_{l=-\infty}^{+\infty} x(l)T[\delta(k-l)]$$

En posant $h_l(k) = T[\delta(k-l)]$ on obtient :

$$y(k) = \sum_{l=-\infty}^{+\infty} x(l)h_l(k) \tag{4.2}$$

On remarque que la réponse impulsionnelle $h_l(k)$ dépend de k et l. Cette dernière relation devient plus maniable si on impose au système la condition d'invariance dans le temps.

2.6 SYSTEME LINEAIRE INVARIANT

Un système linéaire est invariant s'il vérifie la propriété suivante :

si y(n) est la réponse à l'excitation x(n) alors y(n-k) est la réponse à l'excitation x(n-k) .
La relation (4.2) devient :

$$y(k) = \sum_{l=-\infty}^{+\infty} x(l)h(k-l)$$

$$= x(k) \star h(k)$$

$$= \sum_{l=-\infty}^{+\infty} h(l)x(k-l)$$

$$= h(k) \star x(k)$$

$\star$ dénote le produit de convolution

Donc un système linéaire invariant effectue tout simplement le produit de convolution entre le signal d'excitation et sa réponse impulsionnelle (figure 4.1). La transformée en z de la réponse de ce système se calcule facilement en utilisant la propriété :

La transformée en z d'un produit de convolution est le produit des transformées en z :

$$si \quad y(n) = x(n) \star h(n) \quad alors \quad Y(z) = X(z)H(z) \tag{4.3}$$

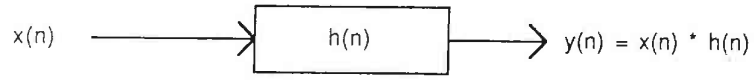


Figure 4.1: Système linéaire invariant

2.7 SYSTEME STABLE

Un système est stable si à chaque exitation bornée correspond une réponse bornée ou encore, si les pôles de la fonction de transfert du système sont à l'intérieur du cercle unité. L'instabilité du filtre [33] se traduit par une oscillation constante ou, une oscillation qui croit de façon exponentielle en réponse à l'impulsion unité.

Un système linéaire invariant est stable si et seulement si [40,26] :

$$S = \sum_{k=-\infty}^{+\infty} |h(k)| < \infty$$

h(k) étant la réponse impulsionnelle du système.

2.8 SYSTEME CAUSAL

Un système est causal, si sa réponse change après le changement de son excitation. Autrement dit, si $x_1(n) = x_2(n)$ pour $n < n_0$ alors $y_1(n) = y_2(n)$ pour $n < n_0$.

Un système linéaire invariant est causal si et seulement si :

$h(n) = 0 \quad pour \quad n < 0$

Ces systèmes caractérisés par le fait que leur réponse ne précède jamais leur excitation, sont les seuls systèmes physiquement réalisables.

2.9 EQUATION AUX DIFFERENCES

Dans un système quelconque le signal d'entrée et le signal de sortie sont liés par des relations mathématiques. L'excitation x(n) et la réponse y(n) d'un large sous-ensemble de systèmes linéaires satisfont une équation aux différences d'ordre p du type :

$$a_0 y(n) + ... + a_p y(n-p) = b_0 x(n) + ... + b_q x(n-q)$$

ou encore :

$$\sum_{i=0}^{p} a_i y(n-i) = \sum_{j=0}^{q} b_j x(n-j) \tag{4.4}$$

Cette équation exprime le fait que la réponse y(n) du système dépend de l'excitation x(n) et des p réponses et des q excitations précédentes (passé et présent).

Une telle équation est appelée équation aux différences linéaire à coefficients constants d'ordre p. Dans cette équation, l'ensemble des coefficients a_i *et* b_j représente le comportement du système à l'instant n. La relation (4.4) est la version discrète des équations différentielles linéaires caractérisant les systèmes linéaires continus ou analogiques.

Les statisticiens appellent ce système un modèle ARMA (modèle autorégressif à moyenne mobile) les coefficients a_i caractérisant la partie AR et les coefficients b_i la partie MA.

La fonction de transfert d'un tel système est donnée par :

$$H(z) = \frac{\sum_{i=0}^{q} b_i z^{-i}}{\sum_{i=0}^{p} a_i z^{-i}}$$

3 MODELISATION DE L'EXCITATION

Pour un modèle autorégressif, la sortie y(n) est une combinaison linéaire des sorties précédentes et d'une entrée u_n :

$$y_n = -\sum_{k=1}^{p} c_k y_{n-k} + G u_n \tag{4.5}$$

En appliquant la transformée en z aux deux membres de cette équation on obtient d'après (4.3) :

$$C(z)Y(z) = GU(z)$$

Ou encore :

$$Y(z) = \frac{G}{C(z)} U(z)$$

Donc ce système a pour fonction de transfert H(z) :

$$H(z) = \frac{G}{C(z)} = \frac{G}{1 + \sum_{k=1}^{p} c_k z^{-k}}$$

G étant le gain du filtre.

La figure 4.2 montre la représentation schématique de ce modèle dans les domaines fréquenciel et temporel.

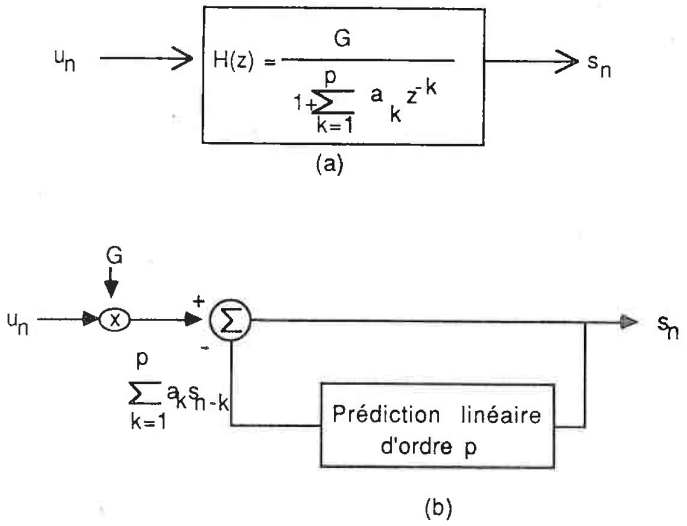


Figure 4.2: représentation fréquentielle et temporelle d'un modèle autorégressif

En comparant la relation (4.5) et la relation (2.3) du deuxième chapitre on remarque que :

1. L'analyse par prédiction linéaire (LPC) peut être vue comme le filtrage d'un signal s(n) par un filtre (filtre inverse du modèle) dont la fonction de transfert est A(z), suivi d'une minimisation (au sens des moindres carrés) de l'erreur e_n (figure 4.3a).

2. Pour retrouver le signal de départ il suffit d'exciter le filtre dont la fonction de transfert est $\frac{1}{A(z)}$ par une entrée u_n proportionnelle au gain ($Gu_n = e_n$) (figure 4.3b).

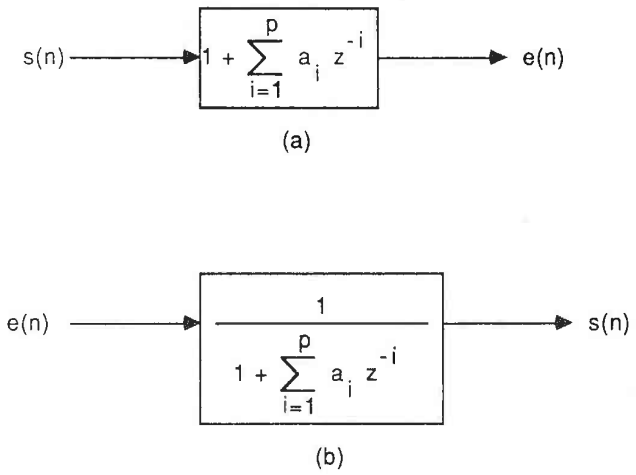


Figure 4.3: Analyse-synthèse

La procédure de Durbin/Levinson permet de calculer les coefficients a_i du modèle, mais il reste à déterminer G.

Pour la détermination du gain G on impose que pour tout signal d'entrée l'énergie du signal de sortie soit égale à l'énergie du signal original.

Cette contrainte est appliquée à deux signaux particuliers : l'impulsion unité et le bruit blanc stationnaire. Dans les deux cas [30,29] l'expression de G est donnée par :

$$G^2 = r(0) + \sum_{k=1}^{p} a_k r(k)$$

d'après l'équation (2.10) on a :

$$G^2 = E_p$$

Le carré du gain G est donc l'erreur de prédiction. Ainsi pour toute fenêtre d'analyse on peut déterminer par la procédure de Durbin / Levinson les coefficients du filtre a_i et le gain G.

3.1 SYNTHESE DE LA PAROLE

Le modèle LPC, estimé par une quelconque méthode de paramétrisation, est figé, le choix de l'excitation du modèle est déterminant dans la qualité de la synthèse obtenue.

Il existe plusieurs techniques pour déterminer l'entrée d'un filtre. Ces techniques proviennent essentiellement de deux modèles :

1. le modèle " bruit blanc + impulsions périodiques " [33].
2. le modèle multi-impulsionnel d'Atal [2,20].

L'objectif commun des différents types d'excitation est de réaliser un compromis entre qualité et débit de transmission.

3.2 MODELE "BRUIT BLANC ET IMPULSIONS PERIODIQUES"

Pour retrouver le signal original (figure 4.3) il suffit d'exciter le filtre dont la fonction de transfert est H(z) par l'erreur e(n). Cette solution demande en plus de la transmission (ou

stockage) des paramètres des filtres la transmission de l'erreur. Dans ce cas précis il est plus économique de transmettre le signal original.

Pour que le codage par LPC soit économique il faut que le débit de transmission soit le plus faible possible. Il faut donc utiliser très peu d'excitations non nulles ou un signal connu d'avance.

Une technique consiste à combiner les deux solutions, mais elle demande la séparation de la parole en sons voisés et non voisés. Le signal d'entrée est constitué de deux sortes d'excitation : un train d'impulsions dont la fréquence est celle de fermeture de la glotte dans le cas voisé et, dans la situation contraire, un bruit blanc permettant de modéliser les sons que produit l'air turbulent (figure 4.4).

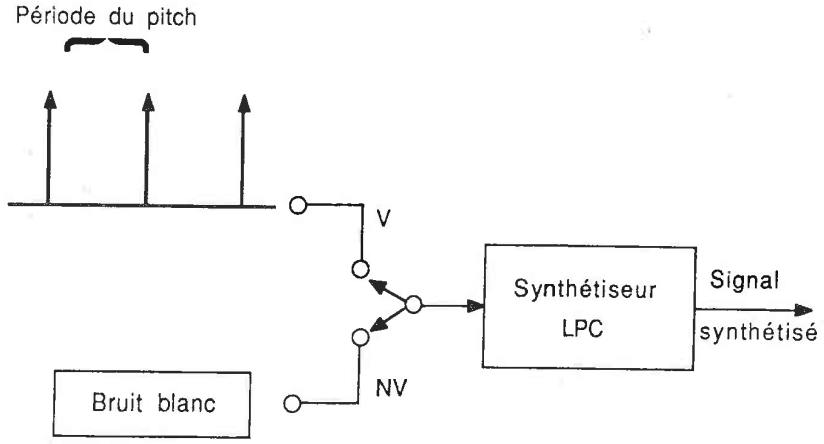


Figure 4.4 Schéma d'un synthétiseur LPC. Les sons voisés sont synthétisés à partir d'un train d'impulsions périodiques et les sons non voisés à partir d'un bruit blanc.

Le schéma de la figure 4.4 montre que si la fenêtre d'analyse (ou trame) est constituée de N points on aura $\frac{N}{I}$ excitations; I étant la période du pitch.

Nous avons vu que le gain G du filtre est calculé de manière à ce qu'il égalise l'énergie du signal d'entrée. Si on représente l'excitation périodique par e(n) on doit avoir :

$$\frac{N}{I}e^2(n) = G^2$$

et donc :

$$e(n) = \begin{cases} G\sqrt{\frac{I}{N}} & \text{si } n = 0, I, 2I, ... \\ 0 & \text{sinon} \end{cases}$$

Si on pré-accentue le signal il faut que la synthèse soit suivie par une étape de post-accentuation.

Le fait d'utiliser une excitation e(n) de moyenne non nulle risque d'introduire un biais pendant l'étape de post-accentuation [33]. Une solution revient à compléter le signal e(n) par un autre signal d'amplitude C, de façon à ce que les deux contributions soient en moyenne nulles :

$$\frac{N}{I}e(n) + C(N - \frac{N}{I}) = 0$$

d'où le nouveau signal de moyenne nulle e(n) :

$$e(n) = \begin{cases} G\sqrt{\frac{I}{N}} & \text{si } n = 0, I, 2I, ... \\ -\sqrt{\frac{I}{N}}\frac{G}{I-1} & \text{sinon} \end{cases}$$

De la même manière il faut que l'énergie du bruit blanc soit égale au carré du gain. Si le bruit blanc g(n) a une variance G_g^2 on doit avoir :

$$Ne^2(n) = \frac{G^2}{G_g^2}g^2(n)$$

et donc l'excitation pour les sons non voisés est donnée par :

$$e(n) = \frac{G}{G_g\sqrt{N}}g(n)$$

Le modèle idéal de la figure 4.4 a été pendant longtemps le seul moyen de synthétiser la parole avec un faible débit de l'ordre de 2400 bits par seconde. Mais les inconvénients d'un tel modèle sont, d'une part, le manque de naturel de la parole synthétisée, d'autre part le manque de robustesse au bruit ambiant.

Le fait que la parole synthétisée manque de naturel est dû essentiellement à :

1. La rigidité du modèle
   Les excitations étant périodiques, à la sortie on récupère un signal périodique manquant de naturel.

2. La séparation voisé, non voisé
   Le problème de détection des sons voisés et non voisés est en pratique très difficile à résoudre surtout dans les zones de transitions où les deux modes d'excitation sont présents.

3. Le choix du train d'impulsions
   Même quand le signal de parole apparaît clairement périodique l'introduction d'un seul point d'excitation durant toute une période fondamentale est une hypothèse trop simplificatrice.

4. La période du pitch
   Pour générer le train d'impulsions il faut connaître avec précision la valeur de la fréquence fondamentale. Les algorithmes qui existent actuellement donnent souvent une valeur moyenne du pitch prise sur plusieurs fenêtres; en plus la fiabilité de ces algorithmes dépend pour beaucoup de la qualité de la parole traitée.

A cause de l'architecture figée du modèle, il est difficile d'améliorer la qualité de la synthèse même en augmentant le débit de transmission ou en changeant la forme d'excitation [52]. Atal et al [2] ont proposé une autre technique d'excitation qui remet complètement en cause le modèle précédent. Cette méthode consiste à remplacer le modèle bruit blanc et impulsions périodiques par un seul modèle où les excitations sont produites de manière à ce que la sortie du filtre "ressemble" le plus possible au signal original.

4 Le MODELE d'ATAL

4.1 MODELE MULTI-IMPULSIONNEL

Comme nous venons de le voir, le découpage du signal de parole en sons voisés et sons non voisés ne permet pas d'obtenir une synthèse de bonne qualité, même en augmentant le débit. Il existe, en effet, des zones pour lesquelles le voisement n'apparaît pas clairement. De plus, même lorsque le signal est nettement périodique l'introduction d'un seul point d'excitation durant toute une période du pitch est souvent insuffisant.

Atal a proposé un modèle d'excitation qui n'utilise ni l'information du voisement, ni un bruit blanc ni la connaissance du pitch. Chaque excitation est caractérisée par sa position et son amplitude. La détermination des excitations est faite de manière à minimiser l'erreur entre le signal original et le signal synthétisé.

Si l'erreur à minimiser est égale à la différence entre le signal synthétique et le signal original le principe d'un tel modèle est illustré par la figure 4.5.

Minimiser la différence (simple ou au carré) entre le signal original et le signal synthétisé n'a pas de sens concret pour l'oreille. Cette erreur est modifiée de façon à tenir compte de la perception humaine.

Si on minimise l'erreur énergétique E :

$$E = \sum_n e^2(n)$$

la plus grande contribution provient des zones ayant une forte énergie et donc la minimisation de cette erreur ne concernerait que ces régions. Pour que cette erreur puisse tenir compte des autres zones Atal a proposé de filtrer l'erreur e(n) par un filtre linéaire. L'effet de ce filtre est de diminuer l'énergie de l'erreur dans les zones formantiques.

Les équations régissant le schéma de la figure 4.5 forment un système non linéaire à 2 M inconnus; M étant le nombre d'excitations. La résolution de ce système, revenant au calcul simultané de toutes les positions et amplitudes, est très complexe. Une façon de résoudre ce système consiste à déterminer à chaque fois la position et l'amplitude d'une excitation (figure 4.6). Ainsi le système à 2 M inconnues est ramené à un système à 2 inconnues (position, amplitude).

L'algorithme de cette méthode, illustré par le schéma de la figure 4.7, consiste, au début à engendrer le signal synthétique sans aucune excitation (juste la mémoire du filtre). On détermine ensuite l'erreur perceptuelle entre la réponse du filtre et le signal original. La minimisation de cette erreur permet de déterminer la position et l'amplitude de la première

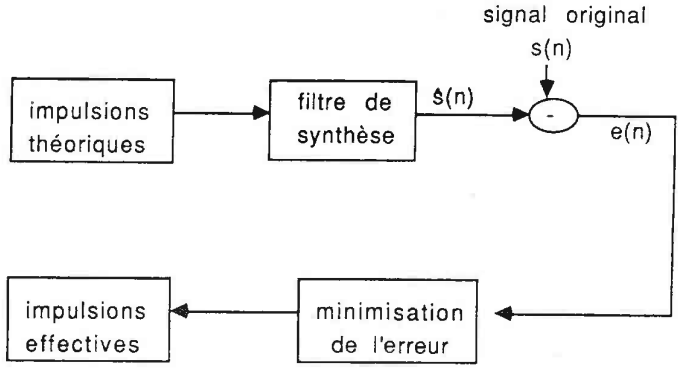


Figure 4.5: Schéma général pour la détermination globale des positions et des amplitudes de l'excitation multi-impulsionnelle.

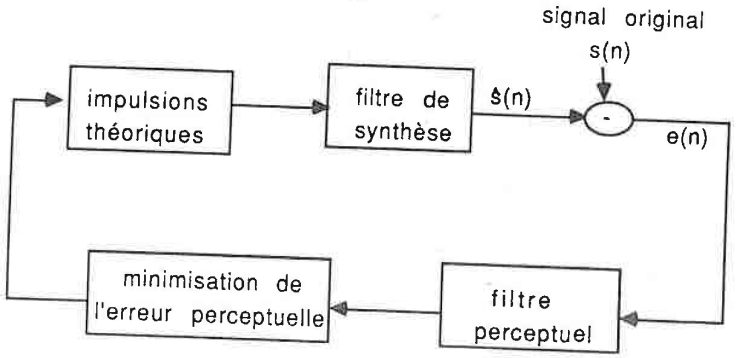


Figure 4.6: Schéma de la procédure d'analyse par synthèse pour déterminer les positions et les amplitudes des impulsions de l'excitation multi-impulsionnelle.

excitation. Une nouvelle erreur perceptuelle est obtenue en retranchant la contribution de l'impulsion qui vient d'être déterminée. Ce processus est réitéré, pour l'estimation des autres impulsions, jusqu'à ce que l'erreur perceptuelle soit inférieure à un seuil donné, ou bien jusqu'à ce que le nombre d'impulsions souhaité soit atteint. L'erreur perceptuelle diminue quand le nombre d'impulsions augmente, mais au delà d'une certaine limite la contribution de nouvelles impulsions est négligeable.

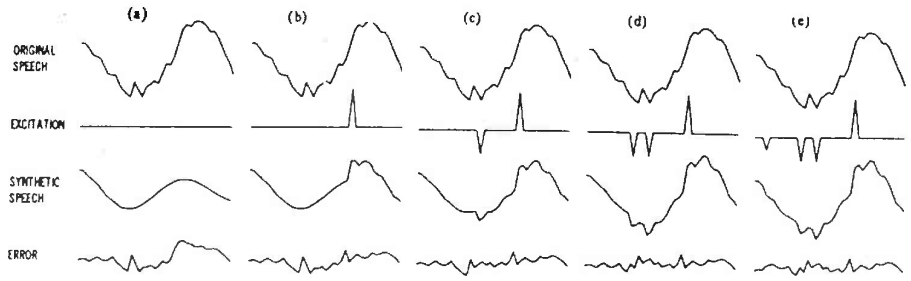


Figure 4.7: Illustration du principe de la méthode. Le signal original, l'excitation, le signal synthétique et les signaux d'erreur sont tracés. (a) au début sans excitation, (b) avec une impulsion, (c) avec deux impulsions, (d) avec trois impulsions, (e) avec quatre impulsions [2].

4.2 FILTRE PERCEPTUEL

Le rôle essentiel de ce filtre est de pondérer l'erreur e(n). Cette pondération est choisie de façon à tolérer une erreur plus grande dans les zones formantiques, c'est à dire les zones qui possèdent une forte énergie, que dans les autres.

Atal a proposé un filtre dont la fonction de transfert est la suivante :

$$P(z) = \frac{\sum_{i=0}^{p} a_i z^{-i}}{\sum_{i=0}^{p} a_i \gamma^i z^{-i}}$$

qui peut s'écrire encore :

$$P(z) = \frac{A(z)}{A^*(z)}$$

où A(z), défini par les coefficients LPC (a_i, $0 \leq i \leq p$) designe le filtre inverse du filtre de synthèse. A^* designe la transformée en z de la séquence des coefficients ($a_i\gamma^i$, $0 \leq i \leq p$). γ est un paramètre déterminé à partir de tests auditifs.

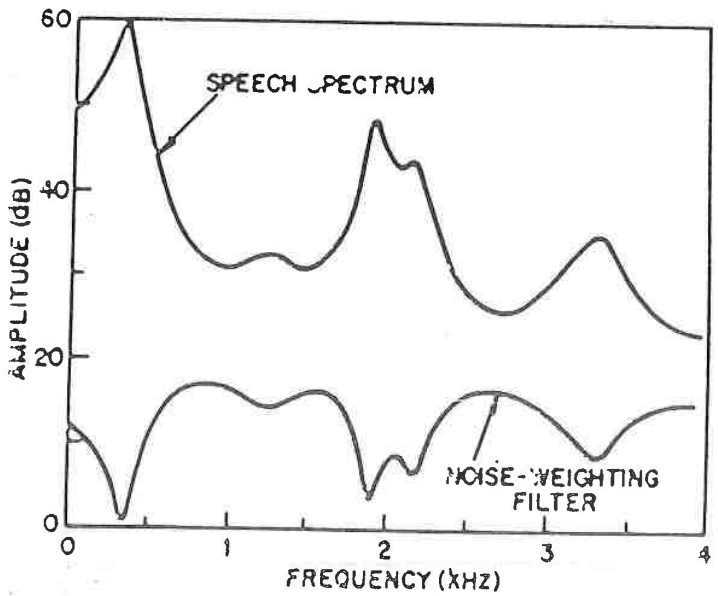


Figure 4.8: exemple du spectre d'un signal de parole et de la réponse fréquentielle du filtre perceptuel associé [2].

Comme nous l'avons signalé précédemment la détermination des excitations se fait une par une. La figure 4.9 explicite le schéma de la figure 4.6.

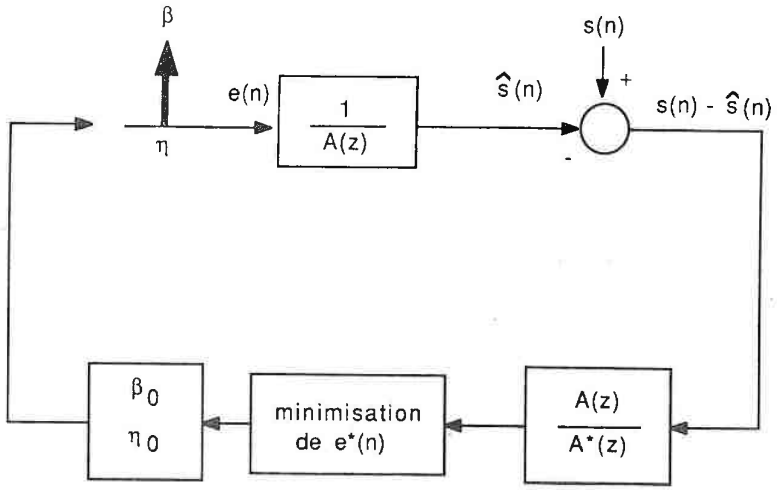


Figure 4.9: Schéma de la procédure d'analyse par synthèse. On suppose qu'il y a une excitation d'amplitude β à la position η.

La détermination d'une excitation revient à exciter le filtre de synthèse par une entrée e(n) d'amplitude β à la position η.

La différence entre le signal original et le signal synthétisé est filtrée par un filtre perceptuel dont la fonction de transfert est :

$$H(z) = \frac{A(z)}{A^*(z)}$$

C'est la minimisation de l'erreur perceptuelle qui fournit les valeurs η_0 *et* β_0 *de* η *et* β.

Les notations utilisées sont :

$$A(z) = 1 + \sum_{i=1}^{p} a_i z^{-i}$$

$$A^*(z) = 1 + \sum_{i=1}^{p} a_i \gamma^i z^{-i} = A(z/\gamma)$$

$$e(n) = \beta\delta(n - \eta)$$

E(z), $E^*(z)$, S(z) et $\hat{S}$ (z) sont respectivement la transformée en z de e(n), $e^*(n)$, s(n) et $\hat{s}$ (n).

$e^*(n)$: erreur perceptuelle
e(n) : signal d'excitation
s(n) : signal original
$\hat{s}(n)$: signal synthétisé

4.3 DETERMINATION DE L'ERREUR PERCEPTUELLE

La figure 4.9 montre qu'on utilise deux filtres :

1. Le filtre de synthèse dont la fonction de transfert est $H(z) = \frac{1}{A(z)}$. et
2. Le filtre perceptuel dont la fonction de transfert est $H(z) = \frac{A(z)}{A^*(z)}$.

L'entrée et la sortie de chaque filtre sont liées par des relations mathématiques. D'après (4.3), l'excitation et la réponse du filtre de synthèse sont liées par :

$$\hat{S}(z) = \frac{E(z)}{A(z)}$$

et du filtre perceptuel par :

$$E^*(z) = \frac{A(z)}{A^*(z)}[S(z) - \hat{S}(z)]$$

en remplaçant $\hat{S}(z)$ par sa valeur on obtient :

$$E^*(z) = \frac{A(z)}{A^*(z)}[S(z) - \frac{E(z)}{A(z)}]$$

ou encore :

$$E^*(z) = [A(z)S(z) - E(z)]\frac{1}{A^*(z)}$$

on pose $A(z)S(z) = R(z)$ la dernière équation devient :

$$E^*(z) = [R(z) - E(z)]\frac{1}{A^*(z)} \tag{4.6}$$

on peut traduire cette dernière relation par le schéma de la figure 4.10.

On désigne par h(n) la réponse impulsionnelle du filtre dont la fonction de transfert est $\frac{1}{A^*(z)}$. En prenant la transformée en z inverse de chaque membre de la relation (4.6) on obtient :

$$e^*(n) = \sum_k h(n-k)[r(k) - e(k)]$$

En remplaçant e(k) par sa valeur on obtient :

$$e^*(n) = \sum_k h(n-k)[r(k) - \beta\delta(k - \eta)]$$

Cette dernière relation peut s'écrire encore :

$$e^*(n) = \sum_k h(n-k)r(k) - \beta h(n - \eta) \tag{4.7}$$

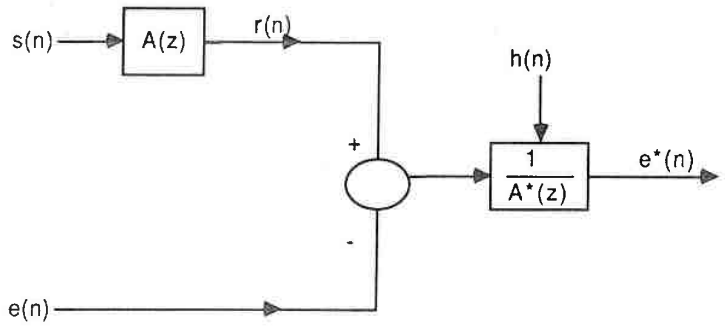


Figure 4.10: schéma du calcul de l'erreur perceptuelle, h(n) est la réponse impulsionnelle du filtre de fonction de transfert $\frac{1}{A^*(z)}$.

Pour simplifier les notations on pose :

$$r^*(n) = \sum_k h(n-k)r(k) \tag{4.8}$$

En reportant la relation (4.8) dans (4.7) on obtient :

$$e^*(n) = r^*(n) - \beta h(n-\eta) \tag{4.9}$$

On minimise l'erreur globale suivante :

$$E = \sum_n e^{*2}(n)$$

$$= \sum_n r^{*2}(n) + \beta^2 \sum_n h^2(n-\eta) - 2\beta \sum_n r^*(n)h(n-\eta) \tag{4.10}$$

En annulant la derivée de E par rapport à β on obtient :

$$\beta = \frac{\sum_n r^*(n)h(n-\eta)}{\sum_n h^2(n-\eta)} \tag{4.11}$$

La première solution consiste à remplacer β par sa valeur dans (4.10) et annuler la dérivée de E par rapport à η. On obtient une équation dont il est difficile de trouver les solutions.

La deuxième solution est une optimisation de la première, elle consiste à remplacer β dans (4.10) et minimiser E.

En remplaçant β dans (4.10) on obtient :

$$E = \sum_n r^{*2}(n) - \frac{[\sum_n r^*(n)h(n-\eta)]^2}{\sum_n h^2(n-\eta)}$$

Comme E est une quantité toujours positive ou nulle, minimiser E revient à maximiser le rapport :

$$\frac{|\sum_n r^*(n)h(n-\eta)|}{\sum_n h^2(n-\eta)}$$

On commence par déterminer la position η qui maximise ce rapport puis on calcule l'amplitude β par la formule (4.11).

Pour la détermination de l'excitation suivante (M-ième excitation) on peut procéder de deux manières.

1. Après avoir retranché la contribution de la nouvelle impulsion qu'on vient de déterminer on calcule la position et l'amplitude de l'impulsion suivante par (4.11).
2. On considère correctes les M positions trouvées précédemment et on recalcule globalement les M amplitudes (β_i, $1 \leq i \leq M$); cela revient à annuler les dérivées partielles de E par rapport à β_i :

$$E = \sum_n e^{*2}(n)$$

La relation (4.9) donne l'erreur perceptuelle correspondant à une seule impulsion, si on a M impulsions cette erreur devient :

$$e^*(n) = r^*(n) - \sum_{i=1}^{M} \beta_i h(n-\eta_i)$$

On pose :

$$a(i,j) = \sum_n h(n-\eta_i)h(n-\eta_j) \quad et$$

$$b_i = \sum_n r^*(n)h(n-\eta_i)$$

En annulant la dérivée de E par rapport à β_i on obtient :

$$\sum_{j=1}^{M} \beta_j a(i,j) - b_i = 0 \quad pour\ tout\ \ i = 1, ..., M$$

Ce qui peut s'écrire sous la forme matricielle suivante :

$$\begin{vmatrix} a(1,1) & a(1,2) & a(1,M-1) & a(1,M) \\ a(2,1) & a(2,2) & a(2,M-1) & a(2,M) \\ & & & \\ & & & \\ & & & \\ a(M,1) & a(M,2) & a(M,M-1) & a(M,M) \end{vmatrix} \cdot \begin{vmatrix} \beta_1 \\ \beta_2 \\ \\ \\ \\ \beta_M \end{vmatrix} = \begin{vmatrix} b_1 \\ b_2 \\ \\ \\ \\ b_M \end{vmatrix}$$

L'utilisation de la deuxième méthode est plus précise mais demande beaucoup de calculs. Une solution intermédiaire consiste à déterminer par la première méthode toutes les positions et amplitudes et résoudre seulement à la fin le dernier système pour la ré-estimation globale des amplitudes.

5 QUELQUES TECHNIQUES DE CODAGE SCALAIRE

Le codage d'un signal quelconque est une nécessité qui apporte beaucoup d'avantages tels qu'une diminution de la redondance de l'information ainsi que des facilités de transmission et de stockage sans oublier les transmissions à caractère confidentiel. L'objectif visé, dans le domaine des communications est de transmettre le signal de parole avec la meilleure qualité possible et le plus faible débit; cela explique l'intérêt des recherches dans ce domaine.

5.1 PCM

La méthode dite PCM (Pulse Code Modulation) est sans aucun doute l'une des plus élémentaires, elle consiste à échantillonner le signal avec une fréquence d'au moins 2 W Hz où W est la fréquence la plus haute contenue dans le signal initial. Chaque amplitude est quantifiée de manière uniforme en l'un des 2^B niveaux, B étant le nombre de bits par échantillon. Le décodage consiste à transformer chaque niveau en amplitude et pour ne pas avoir des fréquences supérieures à W on filtre la séquence des amplitudes par un filtre passe-bas dont la fréquence de coupure est W. La figure 4.11 montre les erreurs introduites par un tel procédé.

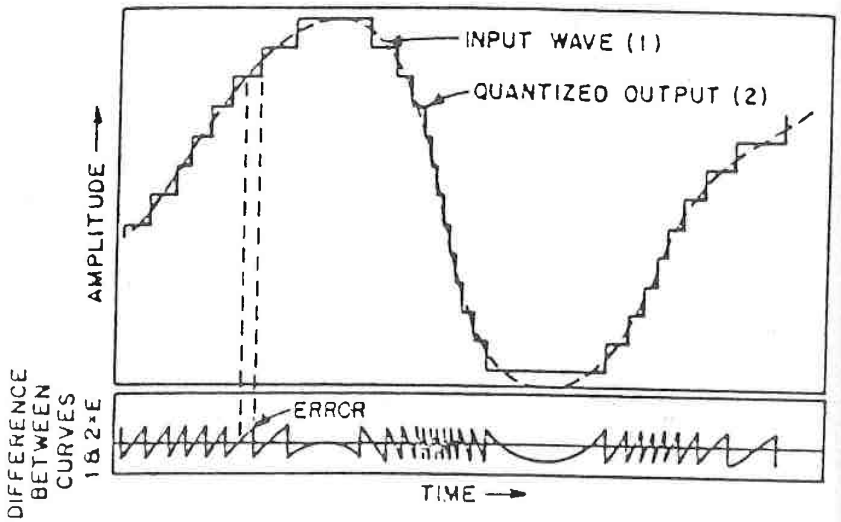


Figure 4.11: Signal original, signal quantifié et l'erreur de quantification par la méthode PCM [23].

Il existe d'autres variantes de cette technique exploitant la corrélation des échantillons et la régularité du signal de parole comme : DPCM (Differencial Pulse-Code Modulation) ou DM (Delta Modulation) [23].

5.2 CODE DE FANO

La caractéristique commune à toutes les techniques de codage qu'on vient de citer est l'introduction d'une erreur de quantification ou distorsion. Il est possible de réduire le débit de transmission sans aucune dégradation du message initial. parmi ces techniques on en citera deux : codages de Fano et de Huffman.

code de Huffman nécessitent une connaissance de la distribution de probabilité du signal à transmettre.

Le principe de base de ce procédé de codage [49] est de représenter les valeurs du signal (ou messages) les moins probables par les éléments les plus longs du code. Ainsi on transmet moins de bits pour les valeurs qui se répètent souvent.

Supposons qu'une source ne puisse produire que 8 messages : A, B, C, D, E, F, G et H. Les probabilités de ces messages sont les suivantes :

A	B	C	D	E	F	G	H
0.1	0.18	0.4	0.05	0.06	0.1	0.07	0.04

La réalisation pratique de ce codage revient à classer les messages par ordre de probabilité décroissante et faire deux groupes de probabilité à peu près égale. Ces groupes sont eux-mêmes divisés en deux sous-groupes et le processus se répète jusqu'à ce que chaque message soit isolé. A chaque division les messages du premier groupe se voient attribuer le symbole 0 et ceux du second le symbole 1 (figure 4.12).

5.3 CODE DE HUFFMAN

Huffman [21] a développé une méthode de codage qui est en général plus efficace que la méthode de Fano. Les deux plus faibles probabilités sont isolées et on attribue à l'une le symbole 0, à l'autre le symbole 1. On regroupe en une seule ces deux probabilités et la probabilité simultanée remplace dans la liste les deux probabilités précédentes. On combine de nouveau les deux probabilités les plus faibles de la liste et ainsi de suite jusqu'à ce que toutes les probabilités aient été combinées (figure 4.13).

Pour obtenir le code correspondant à chaque message, en écrivant de droite à gauche, on part de la gauche du diagramme et on suit la ligne partant de ce message jusqu'à l'extrême droite, en écrivant un 1 ou un 0 comme cela est indiqué à chaque embranchement.

On obtient finalement le code suivant :

Message	C	B	A	F	G	E	D	H
Probabilité	0.40	0.18	0.10	0.10	0.07	0.06	0.05	0.04
Code	1	001	011	0000	0100	0101	00010	00011

Il est à noter que ces deux derniers codes sont des codes sans distorsion (pas d'erreurs

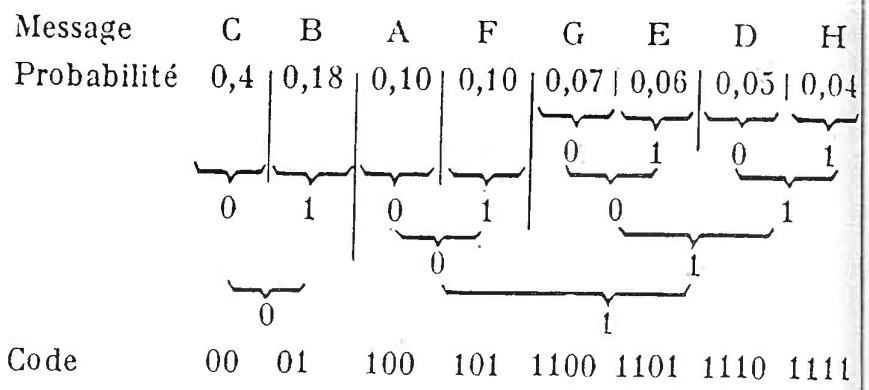


Figure 4.12: schéma illustrant le principe du codage de Fano

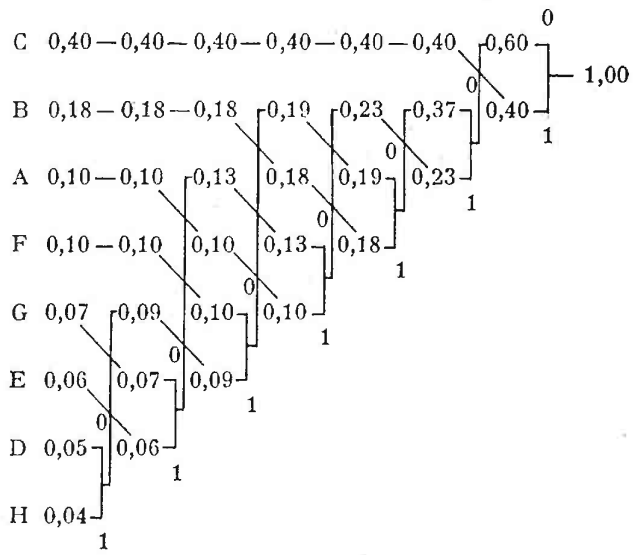


Figure 4.13: Principe du codage de Huffman.

de quantification) et ils vérifient la propriété d'un code à savoir : une succession d'éléments d'un code ne peut être décodée que d'une seule et unique manière et donc on n'a pas besoin d'ajouter, lors de la transmission un signal de synchronisation, cette propriété peut être exprimée par :

Si $x_1x_2....x_n = y_1y_2....y_m$ alors
$x_i = y_i$ et n = m
x_i et y_i sont des éléments du code.

5.4 LPC-10

Cette technique [55] fondée sur la méthode de LPC LPC consiste à transmettre les paramètres d'un nouveau modèle d'ordre 10 toutes les 22.5 millisecondes. A la réception la reconstitution du signal est faite à partir d'un train d'impulsions pour les sons voisés et un bruit blanc pour les sons non-voisés (chapitre 3.3). Cette technique demande donc la transmission :

1. Des coefficients du filtre
2. Des valeur du pitch pour les sons voisés sur 6 bits et la décision du voisement (1 bits)
3. De la valeur du gain
4. D'un signal de synchronisation

Pour des raisons de quantification le filtre est représenté par les coefficients de réflexion $(k_1, k_2, ..., k_{10})$. La répartition des bits alloués à chaque paramètre est la suivante :

	voisé	non voisé
Pitch/voisement	7	7
Gain	5	5
synchronisation	1	1
k(1)	5	5
k(2)	5	5
k(3)	5	5
k(4)	5	5
k(5)	4	
k(6)	4	
k(7)	4	
k(8)	4	
k(9)	4	
k(10)	4	
Total	54	33

La transmission de chaque trame du signal nécessite 54 bits, soit un débit de 2400 bits par seconde.

6 CODAGE VECTORIEL

On remarque que pour LPC-10 sur les 54 bits alloués à chaque trame 41 bits sont utilisés uniquement pour la transmission des coefficients du modèle LPC. En utilisant les techniques de quantification vectorielle on peut coder les paramètres d'un filtre par un indice (exemple 10 bits si on a 1024 prototypes) d'où un gain considérable de bits. En utilisant cette dernière technique on peut transmettre la parole avec un débit de 800 bits par seconde [58].

6.1 CODAGE VECTORIEL "MULTI-ETAGES"

La synthèse de la parole à 800 bits par seconde manque de naturel pour les raisons suivantes : d'une part, la rigidité du modèle et d'autre part, l'erreur de quantification introduite lors du remplacement du vrai modèle par son plus proche voisin. Pour compenser ce manque de précision Gray [19] a proposé un quantificateur à deux ou plusieurs étages.

Le principe de cette méthode consiste à coder un vecteur d'entrée x par le premier ensemble de prototypes, ce qui se traduit par la sélection d'un vecteur y_i. Ensuite une erreur de quantification est évaluée. Cette erreur à son tour est quantifiée par un deuxième ensemble de prototypes et ainsi de suite pour les autres étages (figure 4.14). Finalement le vecteur x est quantifié par :

$$y_i + e_i^1 + e_i^2 + ... + e_i^{p-1}$$

e_i^j est la sortie du quantificateur du j-ème étage.

p étant le nombre d'étages utilisés.

Voici une solution pour générer les ensembles de prototypes à chaque étage : à partir d'un ensemble d'apprentissage de N vecteurs $\{x_i, \ i = 1, \ ... \ , N\}$ et par application d'un algorithme quelconque de classification automatique, on obtient le premier ensemble de prototypes. Ensuite tout le corpus d'apprentissage est quantifié par ces prototypes et pour chaque vecteur une erreur de quantification est évaluée ce qui permet d'élaborer à partir de ces vecteurs d'erreurs un deuxième quantificateur vectoriel et ainsi de suite pour les niveaux supérieurs.

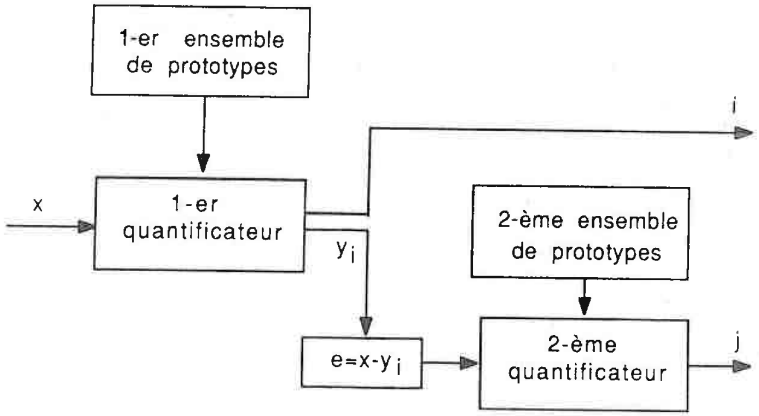


Figure 4.14: Quantification vectorielle à deux étages. Le vecteur x est codé par le premier quantificateur, ensuite l'erreur qui en résulte est quantifiée par un deuxième. Le vecteur x est codé par les deux indices i et j.

7 SYNTHETISEUR PROPOSE

Nous avons vu que la synthèse à partir d'un modèle bruit blanc et impulsions périodiques permet d'atteindre un débit de 2400 et même 800 bits par seconde en utilisant la technique de quantification vectorielle. Cependant la qualité de la synthèse manque de naturel et surtout cette méthode manque de robustesse dans les milieux bruités. Par contre la synthèse multi-impulsionnelle permet d'avoir une synthèse proche de la parole humaine mais le coût de transmission est élevé de l'ordre de 9600 bits par seconde.

Le synthétiseur que nous proposons est fondé sur la quantification vectorielle pour le codage des paramètres du modèle et sur la technique d'Atal pour l'excitation du filtre. Nous avons choisi la quantification vectorielle parce que d'une part la motivation initiale de ce travail était la validation de cette technique et d'autre part parce que la quantification vectorielle permet une grande réduction du débit de transmission. Le choix du modèle d'excitation d'Atal bien que coûteux nous paraît justifié puisqu'il permet de réduire l'erreur de quantification. En effet, si par la règle du plus proche voisin un "mauvais" filtre a été sélectionné alors la méthode multi-impulsionnelle permet de rattraper cette erreur puisque les excitations sont placées là où il faut, de manière à minimiser l'erreur perceptuelle et donc la position des excitations dépend du filtre utilisé.

Nous avons élaboré à partir de plusieurs phrases échantillonnées à 8 Khz un ensemble de prototypes ayant 512 éléments. L'ordre de prédiction utilisé est 10. La fenêtre d'analyse ou trame est de 16 ms ce qui correspond à 128 points du signal échantillonné.

Nous avons utilisé un codage logarithmique pour les amplitudes (5 bits), par contre les positions dont dépend la qualité de la synthèse sont transmises presque directement sur 6 bits (chaque position est codée par le nombre pair le plus proche).

Si on utilise trois excitations pour chaque trame de 16 ms alors la répartition des bits est la suivante :

	8 excitations
amplitudes	48
positions	40
filtre	9
total	75

On utilise 97 bits pour la transmission ou le stockage de chaque trame soit un débit de 6062 bits par seconde.

Les figures (4.15, 4.16) montrent le signal original, le signal d'excitation et le signal synthétisé pour différents débits de transmission. La figure 4.17 montre la synthèse d'un son non voisé.

8 Conclusion

Il est difficile de quantifier les résultats de la qualité de la synthèse. La qualité de la synthèse que nous obtenons dépend du nombre de prototypes et surtout du nombre d'impulsions utilisées par le modèle d'excitation. Le timbre, le rythme, l'intonation sont déterminés par le signal d'excitation. En effet, on a essayé de faire dire à une personne X une phrase prononcée par une personne Y. Les prototypes ont été obtenus à partir de plusieurs phrases prononcées par X. On a synthétisé la phrase prononcée par Y avec ces prototypes. Le signal de sortie correspond aux caractéristiques de la personne Y.
La thèse, récemment publiée, de J. G. Fritsch [14] constitue une étude détaillée sur ce sujet.

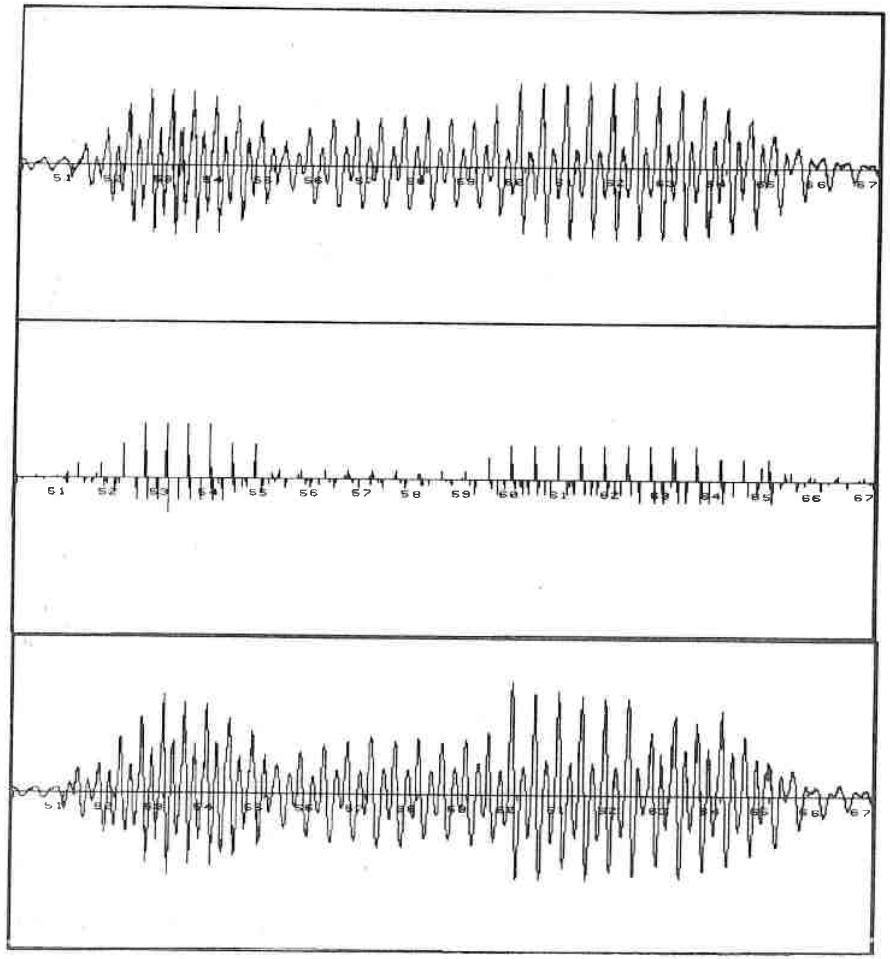


Figure 4.15: Synthèse d'une zone voisée. Signal original, signal d'excitation et signal synthétisé pour un débit de 6062 bits par seconde

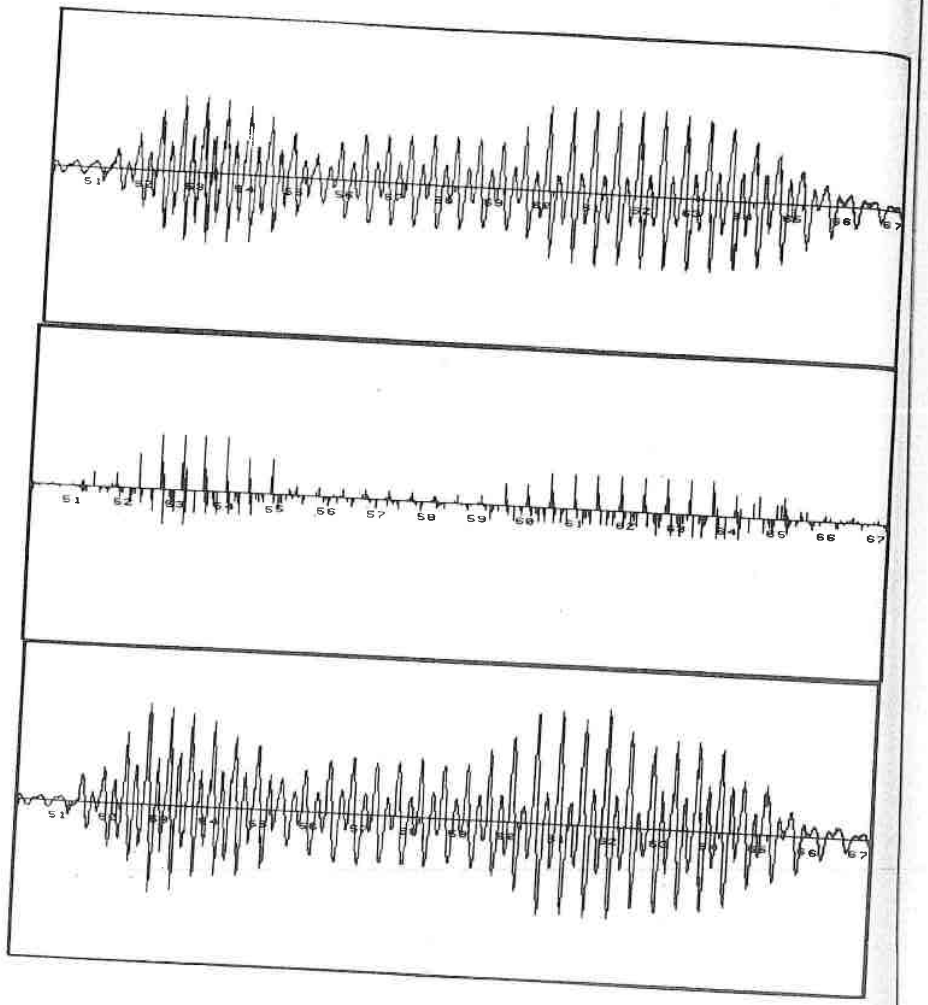


Figure 4.16: synthèse d'une zone voisée. Signal original, signal d'excitation et signal synthétisé pour un débit de 9500 bits par seconde

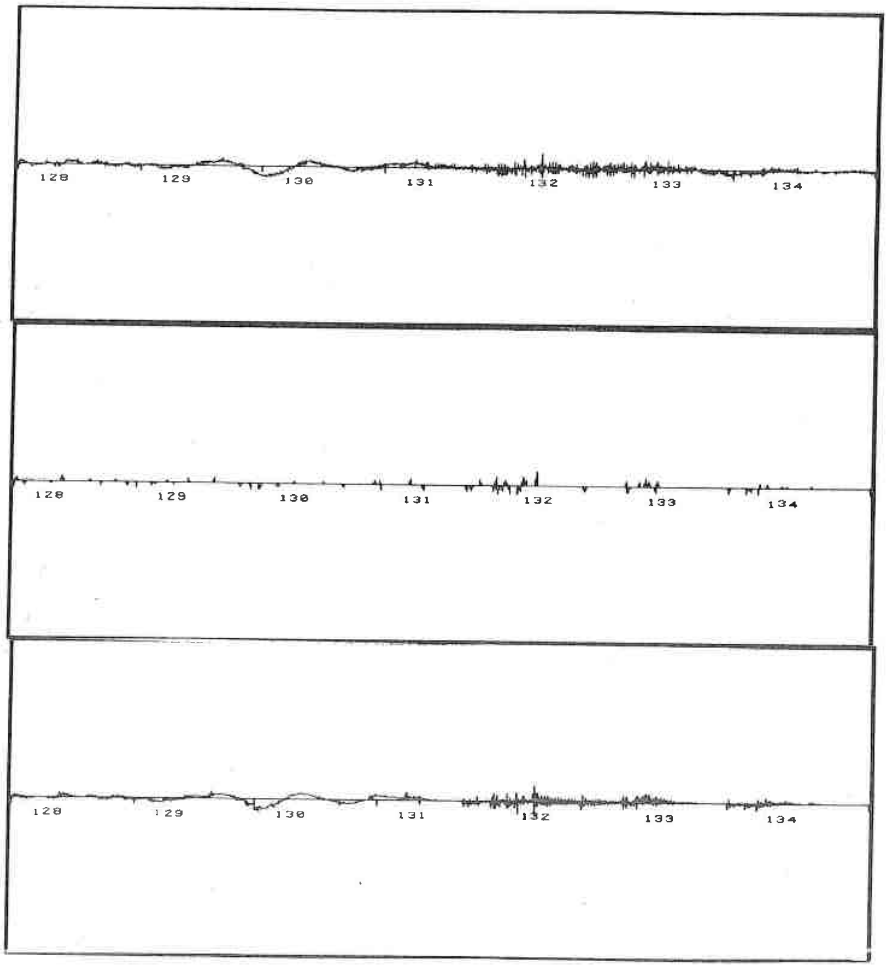


Figure 4.17: synthèse d'une zone non-voisée. Signal original, signal d'excitation et signal synthétisé pour un débit de 6062 bits par seconde

Chapitre 5

RECONNAISSANCE DE LA PAROLE

1 INTRODUCTION

De façon générale la reconnaissance de la parole consiste à transcrire l'onde acoustique en symboles.

La grande complexité de la reconnaissance automatique de la parole provient essentiellement de certaines caractéristiques spécifiques du signal vocal. La parole est caractérisée par une grande variabilité à la fois interlocuteur et intralocuteur. En effet, le conduit vocal présente des différences physiologiques d'un locuteur à un autre. Par ailleurs, les limitations d'ordre mécanique de l'appareil phonatoire conduisent à une forte dépendance contextuelle des sons émis. Les phénomènes articulatoires, le contrôle des cordes vocales, le volume sonore ne sont pas précis. Ils varient en fonction du contexte, l'émotion et la fatigue. Lorsque la vitesse d'évolution est trop élevée, les positions articulatoires normales ne sont pas atteintes. Il en résulte de grandes variations difficiles à analyser.

Pour une même personne, l'évolution d'une phrase ou d'un mot n'est jamais parfaitement répétitive. En effet, la diction fluctue avec l'humeur, l'état de santé, l'environnement dans lequel évolue le locuteur. Ainsi, la vitesse d'évolution, la fréquence fondamentale du signal de parole, l'énergie du son émis et le rythme sont susceptibles d'évoluer de façon significative.

La variabilité est augmentée lorsque l'on passe d'un locuteur à l'autre. Parmi les facteurs introduisant de grandes différences de prononciation entre différents locuteurs on peut citer:

- L'âge
- Le sexe
- L'accent régional

Une autre caractéristique du signal de parole est l'absence de marques de segmentation entre les mots d'un même énoncé. De ce fait, la localisation des mots dans le signal vocal est une opération difficile et indéterministe nécessitant le recours à des connaissances linguistiques.

Comme nous l'avons signalé dans les chapitres précédents la numérisation de l'onde acoustique nécessite un grand débit; ceci est dû en partie au fait que le signal de parole véhicule en plus du message sémantique un grand nombre d'informations telles que les caractéristiques du locuteur (âge, sexe, ...), l'intonation de la voix, l'évolution, le rythme, etc .

Les applications potentielles des systèmes de reconnaissance de la parole sont nombreuses et variées (par exemple machine à dicter automatique [7], le dialogue homme-machine par la voix, etc.). Aussi un système de reconnaissance combiné avec un synthétiseur de parole fournit un système de communication à faible débit.

On peut classer en deux groupes les approches utilisées en reconnaissance de la parole :

- Les méthodes globales et
- Les méthodes analytiques ou phonétiques

Les méthodes globales, limitées en général à la reconnaissance d'un vocabulaire de mots isolés ou enchaînés consistent à comparer globalement le mot inconnu avec tous les mots du vocabulaire. Le mot du vocabulaire qui satisfait au mieux le critère de comparaison avec le mot à identifier est retenu si le taux de ressemblance dépasse un certain seuil. Un inconvénient de ces méthodes est qu'elles nécessitent le stockage de plusieurs références pour chaque mot du vocabulaire et le temps de réponse du système augmente avec la taille du vocabulaire.

Les méthodes analytiques, fondées sur une analyse très fine du signal de parole consistent à représenter chaque mot par une séquence d'unités phonétiques : phonèmes, diphonèmes, syllabes, etc. Les unités sont décrites une fois pour toutes à l'aide des formants, énergie dans les hautes et basses fréquences, passages par zéro etc. Un avantage de ces méthodes est qu'elles ne nécessitent aucun apprentissage, ni le stockage de plusieurs représentants de chaque mot du vocabulaire.

Les méthodes analytiques sont facilement adaptables en reconnaissance de la parole continue et elles sont peut-être les plus satisfaisantes pour ce problème.

Les méthodes stochastiques sont largement utilisées en reconnaissance de mots. Chaque mot est représenté par un automate composé d'états et d'arcs. Le passage d'un état vers un autre se fait selon des lois de probabilité, en particulier, chaque arc reliant l'état i à létat j est caractérisé d'une part par une probabilité de transition de l'état i vers l'état j, et d'autre

part par une probabilité de trouver un spectre donné en suivant cet arc. Ces méthodes sont efficaces surtout en reconnaissance multi-locuteurs [44] et pour le vocabulaire des chiffres français [24].

Nous présentons dans ce chapitre plusieurs algorithmes de reconnaissance fondés sur la quantification vectorielle. Nous proposons diverses variantes d'algorithmes ainsi que l'architecture d'un système de reconnaissance à deux niveaux correspondant à une exploitation optimisée des propriétés de la quantification vectorielle et de la comparaison dynamique : un premier étage, fondé sur la quantification vectorielle, permet de sélectionner rapidement un sous-ensemble de mots candidats qui sont ensuite comparés finement par programmation dynamique au mot inconnu. Nous terminons en proposant une approche de reconnaissance de la parole continue par une méthode globale.

2 RECONNAISSANCE DE MOTS ISOLES

Le fonctionnement d'un système de reconnaissance de mots comprend deux phases : une phase d'apprentissage et une phase de reconnaissance.

Pendant l'apprentissage chacun des mots du vocabulaire de l'application est prononcé isolément par un ou plusieurs locuteurs selon que le système est mono ou multi-locuteur. Les formes acoustiques de chaque mot sont alors mémorisées dans un dictionnaire de formes ou de références.

Lors de la reconnaissance la forme (le mot) inconnue est comparée à toutes les formes de référence qui constituent le vocabulaire. La forme de référence qui aura satisfait au mieux les critères de comparaison (sans dépasser certains seuils de rejet) sera considérée comme le mot reconnu par le système. Dans le cas contraire la forme inconnue sera rejetée. La figure 1 montre schématiquement le principe de ce système de reconnaissance.

Les principales difficultés que l'on rencontre en reconnaissance de la parole par des méthodes globales proviennent essentiellement de la variabilité très importante de certains paramètres caractérisant l'évolution comme :

- Le débit de la parole
- La hauteur de la voix.

La conséquence immédiate de la non-constance de la vitesse d'élocution est l'obtention de formes vocales ayant des longueurs différentes lors d'élocutions d'un même mot par un même locuteur. Ce phénomène est encore accentué par l'état de fatigue du même locuteur ou quand les locuteurs sont différents.

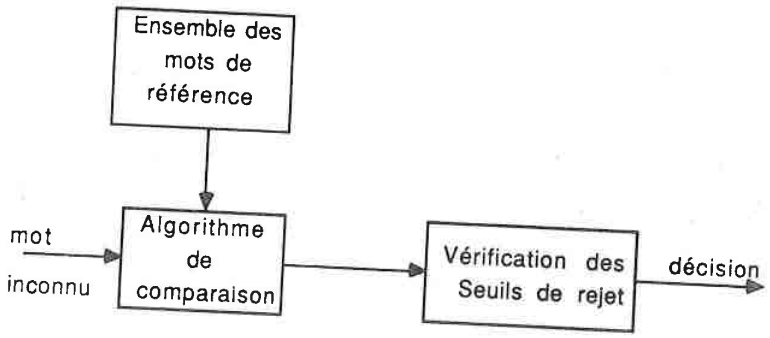


Figure 5.1: Principe général de la reconnaissance de mots isolés.

Pour résoudre le problème de la vitesse d'élocution il y a deux solutions : une normalisation temporelle linéaire ou une normalisation non linéaire.

3 NORMALISATION TEMPORELLE LINEAIRE

Cette technique consiste à imposer une longueur fixe pour toutes les élocutions. Les formes sont normalisées linéairement à une même longueur. On peut opérer une normalisation linéaire par mot, ou par vocabulaire, dans le premier cas la normalisation est variable et dépend de la longueur moyenne de chaque mot du vocabulaire alors que dans le deuxième cas tous les mots du vocabulaire ont la même longueur. Comme on le verra plus loin cette normalisation n'est pas réaliste.

4 NORMALISATION TEMPORELLE NON LINEAIRE

La normalisation temporelle linéaire suppose que tous les sons élémentaires qui composent un mot sont linéairement comprimés ou dilatés en fonction de la vitesse d'élocution. Or la figure 2 montre que la transformation est non linéaire. En effet, alors que certains phonèmes ne sont guère altérés lors d'une évolution rapide, par rapport à une élocution normale, d'autres par contre sont fortement comprimés ou déformés. Les figures 5.2a et 5.2b mettent en évidence les distorsions temporelles apparaissant lors de deux élocutions du mot "chapeau" prononcé par un même locuteur. En comparant celles-ci on observe que la fricative sourde /ch/ (prélèvements 1 à 8 sur 2a et 1 à 4 sur 2b) et la voyelle /a/ (prélèvement 8 à 14 sur 2a et 5 à 9 sur 2b) ont été considérablement comprimées tandis que la plosive /p/ (prélèvement 15 à 20 sur 2a et 10 à 15 sur 2b) n'a subi aucune déformation temporelle.

La représentation spectrographique du mot "chapeau" sur la figure 5.2 est fournie par un vocodeur à 16 canaux couvrant la plage de fréquences 0-5000 hz. Chaque colonne correspond à 20 ms de parole. L'intensité sonore dans chaque canal est codée de 0 à 7, soit 8 valeurs représentées graphiquement par 8 caractères depuis /blanc/ (pour 0) jusqu'à **/%/** (pour 7).

5 DETECTION DES FRONTIERES D'UN MOT

Un problème indépendant de la vitesse d'élocution et dont dépend le résultat du système de reconnaissance est la détection du début et fin de chaque mot. En effet, pour la normalisation linéaire ou non linéaire, il est nécessaire de connaître avec précision les frontières des mots afin d'effectuer la transformation envisagée. Le problème de détection des frontières se pose surtout pour les mots qui commencent ou se terminent par des phonèmes faiblement énergétiques. Ces phonèmes sont difficiles à dissocier du bruit ambiant [27]. Une mauvaise détection des frontières se traduit par une troncature du mot, ce qui peut par la suite fausser la reconnaissance.

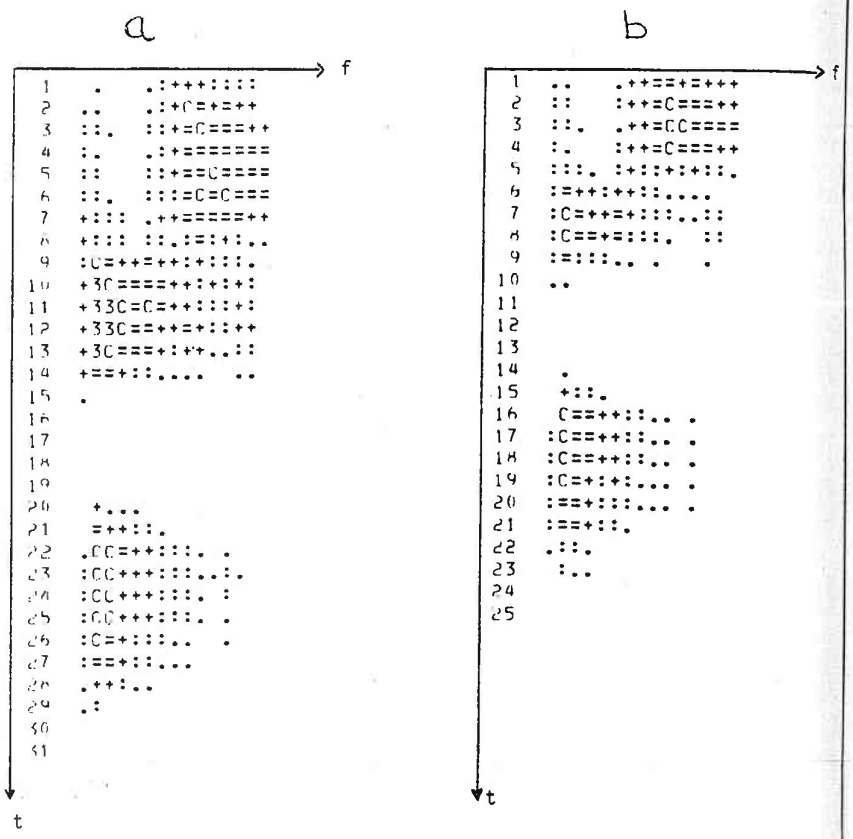


Figure 5.2: Distorsions temporelles apparaissant lors de deux élocutions différentes du mot "chapeau" [34].

6 RECALAGE TEMPOREL

Les techniques de recalage temporel sont utilisées pour compenser les distorsions introduites par les variations de la vitesse d'élocution. Le rôle essentiel de ces techniques est de synchroniser les échelles de temps entre deux formes vocales à comparer (figure 3a).

Les deux formes vocales s'expriment comme deux séquences de vecteurs.

Soit :

$A = a_1, a_2, ..., a_I$ la première forme
$B = b_1, b_2, ..., b_J$ la deuxième forme

où :

- I et J représentent respectivement les longueurs en nombre de fenêtres (ou trames) des formes vocales A et B.
- a_i *et* b_j sont des paramétrisations quelconques de la i-ème trame de la forme A et de la j-ème trame de la forme B.

Comme le montre le schéma de la figure 3b le recalage temporel consiste à trouver le chemin F qui donne la meilleure similitude entre les deux formes. Pour pouvoir différencier deux chemins qui ont la même allure on associe par l'intermédiaire d'une distance un côut ou une mesure de similitude à chaque chemin. Autrement dit, si les deux formes acoustiques A et B sont associées respectivement à un axe i et j du plan de comparaison le recalage temporel se ramène à la recherche d'un chemin F :

F = c(1)c(2) ... c(K)

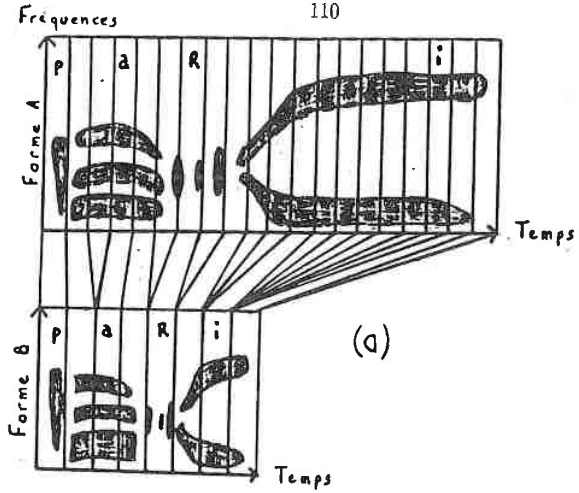
où c(k) = (i(k), j(k)) représente un point de comparaison par lequel passe le chemin F.

F est appelé la fonction de recalage temporel. Quand il n'y a pas de différence entre les deux formes A et B le chemin F coincide avec l'axe diagonal j = i.

A chaque chemin F est associé un coût C(F) :

$$C(F) = \sum_i d(c(k))$$

où $d(c(k)) = d(a_{i(k)}, b_{j(k)})$ est la distance inter-trames.



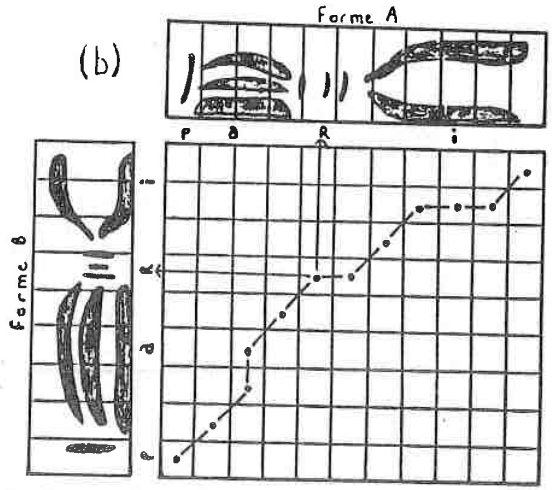


Figure 5.3: Distorsions temporelles observées sur deux réalisations du mot /PARIS/ (a) et recalage temporel appliqué pour compenser les différences de durée et les variations non linéaires du rythme (b) [13].

Pour pouvoir favoriser certaines transitions par rapport à d'autres les distances inter-trames sont pondérées.

Finalement, la distance entre A et B est le coût normalisé associé au chemin optimal :

$$D(A,B) = min_F \frac{\sum_{k=1}^{K} d(c(k)).p(k)}{N(k)} \tag{5.1}$$

où :

- K est le nombre de points du chemin de recalage
- p(k) est la fonction de pondération
- N(K) = $\sum_{k=1}^{K} p(k)$ est un facteur de normalisation dont l'effet est de rendre la distance D(A,B) indépendante de la longueur du chemin de recalage.

Le chemin de recalage tel qu'il a été décrit jusqu'ici est une courbe quelconque pouvant contenir des retours-arrières et des circuits. Pour que le chemin de recalage soit réaliste et tienne compte de l'évolution dans le temps du signal vocal Sakoe et Shiba [Sak-78] ont introduit les contraintes suivantes :

1. Monotonie croissante

$$\begin{cases} i(k-1) < i(k) \\ j(k-1) < j(k) \end{cases}$$

2. Continuité

$$\begin{cases} i(k-1) - i(k) \leq 1 \\ j(k-1) - j(k) \leq 1 \end{cases}$$

Ces deux contraintes montrent que deux points successifs du plan de comparaison c(k-1) et c(k) = (i(k),j(k)) sont liés par les relations :

$$c(k-1) = \begin{cases} (i(k), j(k)-1) \\ (i(k)-1, j(k)-1) \\ (i(k)-1, j(k)) \end{cases}$$

3. Conditions aux frontières

$$\begin{cases} i(1) = 1, j(1) = 1 \\ i(K) = I, j(K) = J \end{cases}$$

Cette contrainte est très importante puisqu'elle peut fausser complètement l'algorithme du recalage temporel si elle n'est pas vérifiée. En effet, en mettant en correspondance les débuts et fins des deux formes à comparer, on suppose implicitement que l'algorithme détermine avec une grande précision les frontières de chaque forme vocale.

4. Domaine de définition du chemin de recalage
   $|i(k) - j(k)| < r$
   où r est un entier positif.

Cette condition permet de restreindre le recalage temporel dans des zones réalistes.

5. Contraintes locales
   Le rôle essentiel de ces contraintes locales est d'empêcher le chemin de recalage d'évoluer toujours dans le même sens horizontalement ou verticalement. Un chemin de recalage vertical ou horizontal se traduit par des compressions ou des dilatations irréalistes.

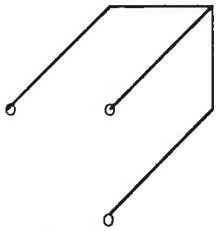
Sakoe et Shiba [51] ont défini une mesure de l'évolution du chemin de recalage par :

$$P = n/m$$

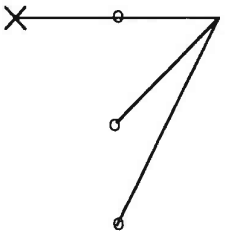
où m représente le nombre de déplacements successifs verticaux ou horizontaux et n le nombre de déplacements successifs diagonaux. Le rapport P est souvent choisi entre 0 et 2. Si $P = \infty$ (m = 0; seuls les déplacements diagonaux sont autorisés) le chemin de recalage coincide avec l'axe diagonal j = i.

Il y a plusieurs façons de choisir les contraintes locales. On interdit par exemple au chemin de recalage d'aller consécutivement deux fois dans la même direction si le sens de déplacement est horizontal ou vertical. Les contraintes de Sakoe et Chiba [51], les contraintes d'Itakura [22] et les contraintes simples sont les contraintes les plus utilisées.

Contrainte simple



Contrainte de Sakoe et Chiba



Contrainte d'Itakura

7 DETERMINATION DES COEFFICIENTS DE PONDERATION

Les coefficients de pondération sont de deux types : symétriques ou asymétriques. Les pondérations utilisées habituellement sont les suivantes :

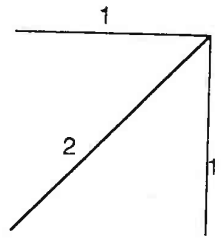
p(k) = (i(k)-i(k-1))+(j(k)-j(k-1)) pour la forme symétrique et

p(k) = i(k)-i(k-1)

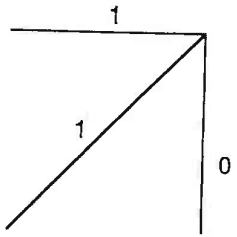
ou

p(k) = j(k)-j(k-1) pour la forme asymétrique.

En appliquant ces pondérations aux contraintes simples on obtient :



Contraintes symétriques



Contraintes asymétriques

Avec ces nouveaux paramètres la relation (5.1) devient :

$$D(A,B) = \frac{1}{N} min_F \sum_{k=1}^{K} d(c(k)).p(k) \qquad (5.2)$$

où

- N = I ou J pour la forme asymétrique et
- N = I + J pour la forme symétrique.

Ainsi, pour toute fonction de recalage F le chemin associé à celle-ci a une longueur constante. On remarque que pour la forme symétrique les déplacements verticaux ne sont pas pris en compte, ce qui implique que certains vecteurs de la forme B pourraient être exclus dans le calcul de la distance D(A,B). Par contre pour la forme symétrique la contribution de tous les vecteurs aussi bien ceux de la forme B que ceux de A est prise en compte dans le calcul de D(A,B).

8 PROGRAMMATION DYNAMIQUE

Après toutes ces simplifications l'équation (5.2) peut être résolue par programmation dynamique à l'aide du principe d'optimalité local introduit par Bellman [3]. Pour les contraintes symétriques on obtient la solution :

g(1,1) = 2d(1,1) (On suppose une transition depuis le point fictif (0,0))

$$g(i,j) = min \begin{cases} g(i,j-1) + d(i,j) \\ g(i-1,j-1) + 2d(i,j) \\ g(i-1,j) + d(i,j) \end{cases}$$

La distance normalisée est donnée par :

$$D(A,B) = \frac{1}{N} g(I,J) \qquad où \;\; N = I + J \qquad (5.3)$$

Ces équations permettent de construire l'algorithme de programmation dynamique suivant :

1. Initialisation : g(1,1) = 2 d(1,1)
2. Evaluation de g(i,j) pour $1 \leq i \leq I$ *et* $1 \leq j \leq J$
3. Détermination de D(A,B) par la relation (5.3)

La programmation dynamique est sans aucun doute la technique qui apporte la meilleure solution au problème de recalage temporel mais la méthode est coûteuse en temps de calcul. Pour le réduire on limite le domaine de recherche du chemin de recalage optimal (figure 4).

A partir de cet algorithme de base de nombreuses versions ont été proposées les unes pour la reconnaissance de mots isolés [42,45,43,36], d'autres pour la reconnaissance de mots enchaînés [50,37,38,48]. On trouve aussi des algorithmes qui résolvent le problème de la détection des frontières de mots par l'introduction de zones floues aux frontières de chaque mot [45]. La thèse de J.Di Martino [34] constitue une étude plus détaillée sur ce sujet. La programmation dynamique est aussi appliquée avec succès pour la reconnaissance de treillis de phonèmes [7].

9 RECONNAISSANCE DE MOTS ISOLES PAR QUANTIFICATION VECTORIELLE

La reconnaissance de mots multi-locuteur par programmation dynamique demande d'une part beaucoup de calculs et d'autre part le stockage de plusieurs références pour chaque mot du vocabulaire. La difficulté qui en résulte est le choix des références à stocker pour chaque mot : il faut que ces références reflètent indépendemment du locuteur toutes les prononciations possibles du même mot. Cette contrainte est en pratique difficilement réalisable puisqu'elle demande le stockage de beaucoup de références pour plusieurs locuteurs. La quantification vectorielle fournit une solution au problème du stockage des références. En effet, la QV donne une représentation statistique de chaque mot du vocabulaire. Cette représentation, obtenue à partir de plusieurs répétitions de chaque mot du vocabulaire, est

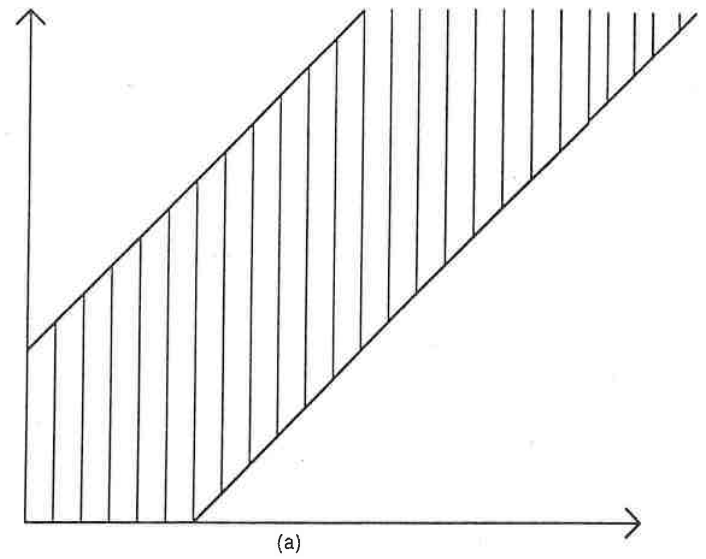


Figure 5.4: Limitation des zones de recherche du chemin de recalage. (a) domaine proposé par Sakoe et Chiba [51] et (b) domaine proposé par Myers [36]

assez fine car les prototypes sont de l'ordre d'une vingtaine de millisecondes, en plus ces prototypes sont indépendants les uns des autres.

9.1 DESCRIPTION DE LA BASE DE DONNEES

La base de données utilisée pour les expériences de test est composée des 10 chiffres de 0 à neuf, prononcés par 24 locuteurs (12 masculins et 12 féminins), chaque locuteur ayant prononcé 4 fois le vocabulaire. Tous les mots sont quantifiés sur 12 bits. La fréquence d'échantillonnage utilisée est de 12 Khz. La paramétrisation choisie du signal est la méthode de prédiction linéaire (LPC) d'ordre 16. La distorsion utilisée aussi bien pour l'apprentissage que pour la reconnaissance est celle d'Itakura-Saito à gain normalisé. Chaque fenêtre d'analyse ou trame est une suite de 128 points soit 10,66 ms. A chaque trame est appliquée une pré-accentuation de 0,94 puis une fenêtre de Hamming. Toutes les méthodes de reconnaissance sont évaluées d'abord sur le corpus masculin, puis sur le corpus féminin et enfin sur la totalité de la base. Pour le corpus féminin et masculin les six premières personnes servent pour l'apprentissage du système et les six autres pour le test. Pour l'évaluation de la méthode sur toute la base les douze premières personnes (6 féminins et 6 masculins) servent pour l'apprentissage.

La détection de début et fin de mot aussi bien pour l'apprentissage que pour la reconnaissance est faite automatiquement : à partir d'un grand échantillon on garde 8 prototypes représentatifs du bruit ambiant de la salle d'enregistrement. Chaque trame du signal à analyser est comparée à ces prototypes et la trame dont la distorsion dépasse un seuil S donné est considérée comme étant le début du mot. La détection de fin de mot est faite de manière similaire. Il est à noter que le succès des méthodes globales dépend entièrement du résultat du détecteur, la plupart des erreurs commises par le système provenant de mauvaises détections des frontières de mots.

9.2 UN SYSTEME DE RECONNAISSANSE A BASE DE QV

Un tel système est schématisé par la figure 5. Chaque ensemble de prototypes est obtenu par QV à partir de plusieurs répétitions du même mot. Le mot inconnu est comparé à tous les ensembles de prototypes et sera identifié au mot du vocabulaire dont l'ensemble de prototypes fournit la meilleure similitude.

Un mot inconnu X peut être exprimé comme une suite de vecteurs x_i, les x_i représentant une paramétrisation quelconque de la trame X_i du mot X :

$$X = x_1 x_2 .... x_N$$

N étant le nombre de trames constituant le mot X.

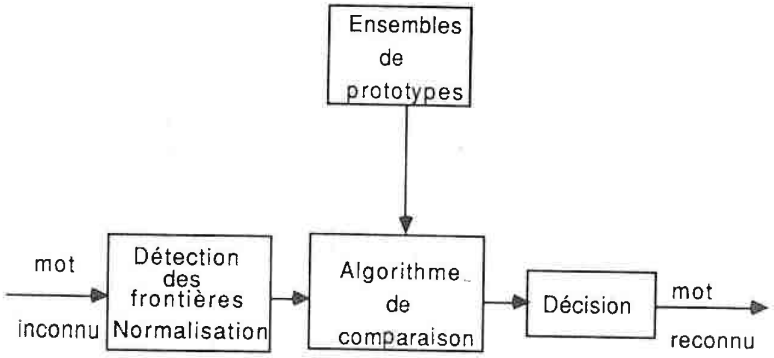


Figure 5.5: Principe général d'un système de reconnaissance de mots isolés à base de quantification vectorielle

La distorsion moyenne correspondant au codage du mot X par l'ensemble de prototypes C_k est :

$$D(X, C_k) = \frac{1}{N} \sum_{j=1}^{N} min_i d(x_j, c_{ki})$$

d étant une mesure quelconque
$C_k = \{c_{k1}, c_{k2}, ..., c_{kM}\}$ l'ensemble de prototypes représentant le mot k du vocabulaire

Le mot inconnu X sera affecté à la classe du mot r du vocabulaire, si :

$$\begin{cases} D(X, C_r) = min_k\ D(X, C_k) \\ D(X, C_r) < S \end{cases}$$

L' introduction du seuil S permet d'éviter que le système ne prenne des décisions dans des situations ambiguës.

Les résultats de cette première méthode de reconnaissance appliquée à notre base de données sont fournis dans la table 1.f pour le corpus féminin, la table 1.m pour le corpus masculin et la table 1.mf pour la totalité du corpus. Chaque mot du vocabulaire est représenté dans ces expériences par 16 prototypes.

Locuteurs masculins		Locuteurs féminins	
dc	95	cf	85
pd	100	da	75
jp	90	ab	92
fl	92.5	cb	85
fa	97.5	nc	72.5
pn	85	bw	92.5
Total	93.33%		83.75%
Table 1.m		Table 1.f	

Locuteurs mixtes			
dc	95	cf	95
pd	97.5	da	75
jp	95	ab	92.5
fl	95	cb	87.5
fa	90	nc	80
pn	80	bw	92.5
Total	92.08%		87.08%
Moyenne	89.58%		
Table 1.mf			

On peut penser que l'augmentation du nombre de prototypes par mot du vocabulaire pourrait améliorer le score final. Les tables 1.2.f et 1.2.m montrent le résultat du système de reconnaissance avec 32 prototypes pour chaque mot du vocabulaire. On remarque que l'augmentation du nombre de prototypes ne se traduit pas par une amélioration significative du score final.

Locuteurs masculins		Locuteurs féminins	
dc	95	cf	82.5
pd	97.5	da	77.5
jp	87.5	ab	92.5
fl	97.5	cb	85
fa	100	nc	80
pn	77.5	bw	95
Total	92.5%		85.41%
Table 1.2.m		Table 1.2.f	

Cette première méthode ne nécessite aucun alignement temporel (contrairement aux techniques qu'on verra par la suite), ni aucune normalisation des mots et donne des résultats satisfaisants. L'inconvénient de cette méthode est qu'elle fait intervenir dans le calcul de la distorsion D(A,B) des comparaisons inutiles. En effet, les prototypes qu'on obtient par QV ne sont pas ordonnés et chaque trame du mot inconnu est comparée à tous les prototypes, aussi bien ceux qui proviennent du début de mot que ceux qui proviennent de la fin des répétitions de chaque mot du vocabulaire.

Pour réduire le nombre de calculs inutiles, il est nécessaire d'établir un ordre de comparaison tel que nous le proposons dans la deuxième méthode exposée ci-après.

9.3 METHODE UTILISANT UN ORDRE DE COMPARAISON

Cette méthode consiste à considérer un mot comme une suite de sections, chaque section étant formée de n trames. Les comparaisons se font section par section ce qui réduit énormément le nombre de calculs.

- CONSTRUCTION DE L'ENSEMBLE DE PROTOTYPES POUR LE MOT K DU VOCABULAIRE

  Cette construction s'appuie sur la méthode de Burton et al [54,5], dont le principe est le suivant :
  Soient $r_1, r_2, ..., r_T$ des répétitions du même mot k du vocabulaire. On découpe toutes ces répétitions en sections de longueur fixe n. Chaque section est une suite de n trames.

  En classifiant à l'aide de l'algorithme présenté au chapitre 3 (paragraphe 9) toutes les premières sections de chaque répétition on obtient le premier ensemble de prototypes

C_{k1} , et ainsi de suite pour les autres sections. On obtient finalement l'ensemble de prototypes C_k du mot k :

$$C_k = \{C_{k1}, C_{k2}, ..., C_{kS_k}\}$$

$$C_{kj} = \{c_{kj1}, c_{kj2}, ..., c_{kjL}\}$$

S_k étant le nombre de sections du mot k et
L le nombre de prototypes pour chaque section

En fait, pour la génération de l'ensemble de prototypes, il y a au moins deux approches possibles, soit un alignement à gauche, soit une normalisation linéaire à une même longueur de toutes les répétitions du même mot.

- RECONNAISSANCE D'UNE FORME INCONNUE
Comme lors de l'apprentissage, le mot inconnu est découpé en sections de même longueur. On compare ensuite chaque section à la section correspondante de chaque mot du vocabulaire. Plus précisément :
Soit à reconnaître la forme X décrite par une suite de vecteurs x_i :

$$X = x_1 x_2 x_3 ..... x_N$$

N étant la longueur (en trames) de la forme inconnue.

Pour simplifier les notations considérons uniquement la comparaison de X avec le mot k du vocabulaire dont l'ensemble de prototypes associé est C_k.

La distorsion correspondant au codage de la trame x_n par la j-ème section C_{kj} du mot k est :

$$d_{mj} = min_i \; d(x_m, c_{kji})$$

d étant la distance inter-trames.

La distorsion pour coder la j-ème section $(x_{u(j)}, ..., x_{v(j)})$ de la forme inconnue par C_{kj} est alors :

$$d(j, C_{kj}) = \sum_{m=u(j)}^{v(j)} d_{mj}$$

La distorsion moyenne pour coder la forme inconnue par l'ensemble de prototypes C_k est finalement :

$$D(X, C_k) = \frac{1}{N} \sum_{j=1}^{w(N)} d(j, C_{kj})$$

w(N) étant le nombre de sections de la forme X.

La forme X sera reconnue comme étant le mot r du vocabulaire si :

$$\begin{cases} D(X, C_r) = min_k \; D(X, C_k) \\ D(X, C_r) < S \end{cases}$$

Pour l'évaluation de cette deuxième méthode nous avons choisi une normalisation linéaire de tous les mots à une longueur fixe $\overline{N}$. Avant de donner les résultats de cette méthode nous avons testé l'influence de la normalisation linéaire. Les tables 1.3.f et 1.3.m représentent les résultats des tables 1.f et 1.m, c'est à dire les résultats de notre première méthode décrite au paragraphe 9 après normalisation de tous les mots.

Locuteurs masculins		Locuteurs féminins	
dc	95	cf	85
pd	97.5	da	77.5
jp	85	ab	92.5
fl	95	cb	82.5
fa	100	nc	82.5
pn	90	bw	92.5
Total	93.75%		85.41%
Table 1.3.m		Table 1.3.f	

Comme on peut le constater le taux de reconnaissance a été amélioré, surtout pour les locuteurs féminins. Ceci s'explique par le fait que les mots du vocabulaire sont courts et, par suite, en moyenne, les variations de longueur de tous les mots du vocabulaire restes faibles.

Les tables 2.f, 2.m et 2.mf représentent les résultats de la reconnaissance multi-sections où tous les mots sont normalisés à une même longueur ($\overline{N} = 60$). Chaque section est une suite de 6 trames successives. Pour la reconnaissance, 8 prototypes ont été utilisés pour

représenter chaque section.

Locuteurs masculins		Locuteurs féminins	
dc	95	cf	90
pd	97.5	da	75
jp	90	ab	92.5
fl	95	cb	82.5
fa	97.5	nc	80
pn	100	bw	87.5
Total	95.83%		89.16%
Table 2.m		Table 2.f	

Locuteurs mixtes			
dc	90	cf	85
pd	97.5	da	80
jp	90	ab	82.5
fl	85	cb	95
fa	97.5	nc	85
pn	92.5	bw	90
Total	92.5%		86.25%
Moyenne	89.37%		
Table 2.mf			

L'avantage de cette méthode par rapport à la première est que pour pratiquement le même score, le nombre de comparaisons a diminué de moitié. En effet, pour la première méthode chaque mot de référence est représenté par 16 prototypes et pour la deuxième chaque mot est une suite de sections. Chaque section est représentée par 8 prototypes. Par conséquent chaque trame du mot inconnu nécessite 16 comparaisons pour la première alors que pour cette méthode elle n'en demande que 8 pour chaque mot du vocabulaire, pour le même taux de reconnaissance.

Dans la deuxième méthode nous avons normalisé tous les mots du vocabulaire à une longueur fixe. Ceci ne serait ni possible ni réaliste pour un autre vocabulaire dont la longueur des mots diffèrerait beaucoup. Dans ce but nous avons repris la deuxième méthode en opérant

des normalisations variables. Les valeurs sont obtenues par moyennage des longueurs des répétitions de chaque mot du vocabulaire.

Lors de la reconnaissance, pour chaque mot du vocabulaire de longueur moyenne N_i, le mot inconnu est normalisé linéairement à cette même longueur puis un calcul de score est évalué entre ce mot inconnu et l'ensembe de prototypes représentant le mot du vocabulaire. Pour que ce score soit indépendant de la longueur on le normalise par N_i.

Cette méthode permet de gagner quelque comparaisons car les mots du vocabulaire n'ont pas tous la même longueur. Par contre, elle nécessite autant de normalisations du mot inconnu que de mots dans le vocabulaire. En général cette dernière opération ne demande pas beaucoup de temps.

Les résultats de cette méthode sont données dans la table 2.2.f pour le corpus féminin, la table 2.2.m pour le corpus masculin et la table 2.2.mf pour la totalité du corpus. Pour pouvoir comparer les résultats, chaque section est également représentée par 8 prototypes.

Locuteurs masculins		Locuteurs féminins	
dc	100	cf	95
pd	100	da	90
jp	90	ab	92.5
fl	97.5	cb	100
fa	100	nc	95
pn	100	bw	100
Total	97.91%		95.41%
Table 2.2.m		Table 2.2.f	

Locuteurs mixtes			
dc	100	cf	92.5
pd	100	da	90
jp	90	ab	92.5
fl	95	cb	97.5
fa	100	nc	90
pn	92.5	bw	100
Total	96.25%		93.75%
Moyenne	95%		

Table 2.2.mf

On remarque que même dans le cas de notre vocabulaire le score de reconnaissance a augmenté de façon significative. Ceci permet de conclure que la normalisation variable est à utiliser même si les mots du vocabulaire ont des longueurs assez proches.

Nous avons montré [11] que pour la reconnaissance mono-locuteur un seul prototype est suffisant pour représenter une section. Par moyennage de toutes les trames formant la section, on arrive à n'effectuer qu'un seul calcul de distance pour chaque section. En reconnaissance multi-locuteur, et par rapport à la méthode précédente, au lieu de traiter chaque trame d'une section on ne traite que le vecteur obtenu par moyennage de toutes les trames constituant la section.

Les résultats de cette variante sont données dans la table 2.3.f pour le corpus féminin, la table 2.3.m pour le corpus masculin et la table 2.3.mf pour la totalité de la base. Chaque section étant représentée par 8 prototypes.

Locuteurs masculins		Locuteurs féminins	
dc	95	cf	97.5
pd	97.5	da	95
jp	90	ab	92.5
fl	97.5	cb	97.5
fa	100	nc	95
pn	100	bw	100
Total	96.66%		96.25%
Table 2.3.m		Table 2.3.f	

Locuteurs mixtes			
dc	97.5	cf	95
pd	100	da	90
jp	90	ab	92.5
fl	97.5	cb	87.5
fa	100	nc	95
pn	100	bw	100
Total	97.5%		93.33%
Moyenne	95.41%		

Table 2.3.mf

L'avantage de cette méthode est que, en plus d'une légère augmentation du score de reconnaissance par rapport à la méthode précédente, on obtient une grande réduction du nombre de calcul de distances. Si chaque section est formée de n trames le nombre de comparaisons est divisé par n. Pour le cas particulier de notre paramétrisation LPC, on evite ainsi (n-1) résolutions d'un système d'équations linéaires à p inconnues.

9.4 RECONNAISSANCE PAR PROGRAMMMATION DYNAMIQUE (dtw)

Afin de comparer les performances des méthodes fondées sur la quantification vectorielle nous avons répété la même expérience en utilisant une méthode de programmation dynamique. Les tables 3.f, 3.m et 3.mf montrent les résultats de la reconnaissance. A cha-

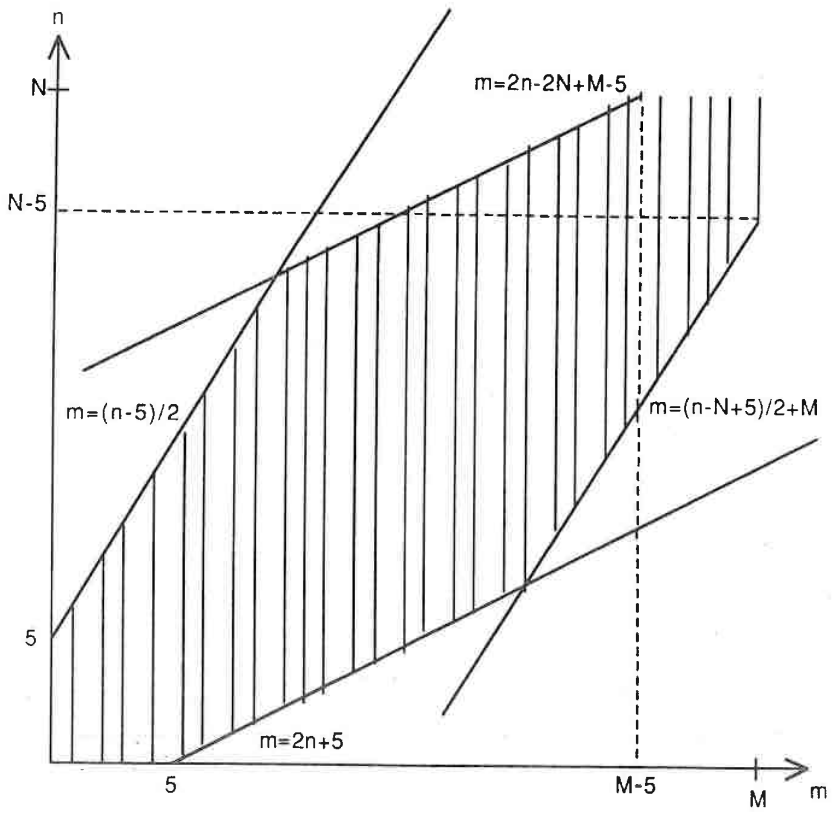


Figure 5.6: Zone de recherche du chemin de recalage optimal

que fois on a utilisé 12 références pour chaque mot du vocabulaire.

Cette méthode donne de bons résultats mais elle reste handicapée d'une part par la place mémoire qu'elle occupe pour stocker les différentes références pour chaque mot du vocabulaire (ce nombre peut être diminué par des méthodes de clustering [43]) et d'autre part le nombre de calcul de distances. Pour minimiser le nombre d'opérations on a restreint la recherche du chemin optimal à la zone indiquée par la figure 6.

Locuteurs masculins		Locuteurs féminins	
dc	100	cf	92.5
pd	100	da	97.5
jp	95	ab	100
fl	100	cb	82.5
fa	100	nc	100
pn	100	bw	100
Total	99.16%		95.41%
Table 3.m		Table 3.f	

Locuteurs mixtes			
dc	100	cf	87.5
pd	97.5	da	92.5
jp	95	ab	97.5
fl	95	cb	95
fa	100	nc	100
pn	100	bw	100
Total	97.91%		95.41%
Moyenne	96.66%		
Table 3.mf			

9.5 COMPARAISON DE DTW ET QV

Sans aucune contrainte sur le chemin de recalage, l'algorithme de comparaison dynamique nécessite l'évaluation de toute la matrice soit M.N comparaisons.

M et N (figure 7) étant les longueurs (en trames) des deux formes à comparer.

En limitant la recherche du chemin de racalage à la zone définie par la figure 6, le nombre de comparaisons est :

ncdtw = M.N-2(S1+S2+S3)

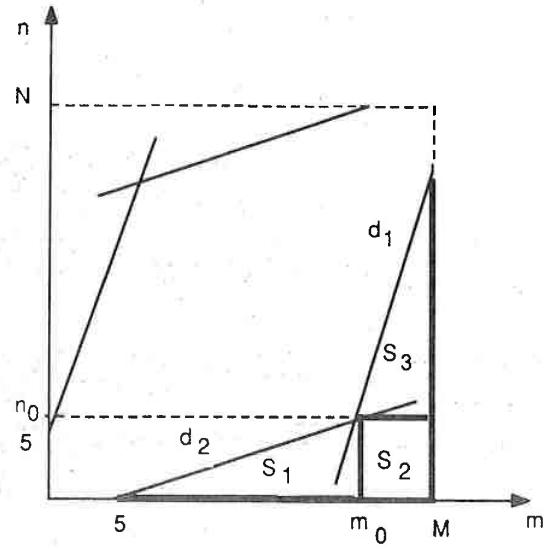


Figure 5.7: Zone de recherche du chemin de recalage optimal utilisée

Pour simplifier le problème et donner juste un ordre de grandeur on suppose que tous les mots du vocabulaire ont une longueur n et donc :

M=N=n et S1=S3

Après ces simplifications on a :

$ncdtw = n^2 - 2(S2 + 2S1)$

Les deux droites d1 et d2 (figure 7) se coupent en (m_0, n_0) :

$$\begin{cases} m_0 = \frac{2n+5}{3} \\ n_0 = \frac{n-5}{3} \end{cases}$$

A partir de ce point on calcule facilement les surfaces S1 et S2, on obtient :

$$ncdtw = \frac{4n^2 + 50n - 125}{9} \cong \frac{4}{9}n^2 \cong \frac{1}{2}n^2$$

Donc, l'utilisation de ces contraintes permet de diminuer à peu près de moitié le nombre de comparaisons. Cependant cet algorithme reste en $o(n^2)$.

Pour reconnaître un mot il faut effectuer :

$\frac{n^2}{2}.V.R$ *comparaisons*

V étant la taille du vocabulaire et
R le nombre de références pour chaque mot du vocabulaire.

Par contre, en utilisant l'algorithme fondé sur la quantification vectorielle multi-sections la reconnaissance du même mot nécessite seulement :

n.V.P comparaisons

P étant le nombre de prototypes représentant chaque section.

Un calcul similaire permet de mettre en évidence le gain en place mémoire d'où l'intérêt des méthodes fondées sur la quantification vectorielle.

9.6 COORDINATION DE DEUX METHODES DE RECONNAISSANCE

1. PRINCIPE
   Malgré la restriction de recherche du chemin optimal l'obtention des résultats des tables 3.f, 3.m et 3.mf demande un temps important. Par contre, les méthodes fondées sur la quantification vectorielle sont rapides et moins précises par rapport aux méthodes de programmation dynamique. Une solution permettant de gagner en rapidité et en précision consiste à faire coopérer ces deux méthodes de reconnaissance.

   La première méthode est l'une parmi celles fondées sur la quantification vectorielle. Dans cette méthode le mot inconnu est comparé à une suite de prototypes reflètant

les caractéristiques spectrales du mot. On introduit ainsi implicitement une erreur de quantification. Pour cette raison, la deuxième méthode travaille directement sur des références non quantifiées. Cette méthode permet d'affiner la décision de la première, mais demande le stockage de plusieurs références pour chaque mot du vocabulaire. Par la suite, on appellera pré-processeur la première méthode et post-processeur la deuxième comme l'indique la figure 8.

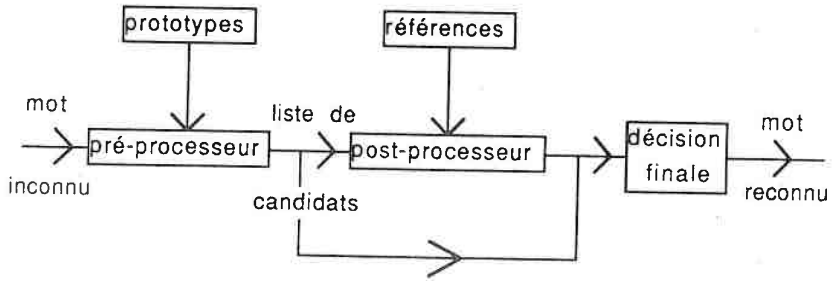


Figure 5.8: Architecture d'un système de reconnaissance à deux étages (le post-processeur devient inutile quand le pré-processeur transmet un seul candidat).

2. CHOIX DU POST-PROCESSEUR
Le post-processeur peut être soit :

(a) Le pré-processeur lui-même, après remplacement des prototypes par de vraies

références.

(b) Un modèle de programmation dynamique (DTW)

3. REGLES GERANT LE COMPORTEMENT DES DEUX PROCESSEURS

(a) Pré-processeur
Soit η le mot du vocabulaire réalisant la plus petite distorsion :

$$D_\eta = min_i\ (D_i)$$

et β le mot réalisant la seconde plus petite :

$$D_\beta = min_{i \neq \eta}\ (D_i)$$

Deux cas peuvent se présenter :

- Règle1 (un seul candidat η)

$$\begin{cases} D_\eta < s_1 \\ D_\beta - D_\eta > s_2 \end{cases}$$

La première condition teste l'appartenance du mot inconnu au vocabulaire. La deuxième permet la prise en compte de l'ecart entre le premier et le deuxième candidat.
- Règle2 (plusieurs candidats)

La liste contiendra tous les candidats j tel que :

$$D_j - D_\eta < s_3$$

(b) Post-processeur
Le post-processeur doit agir en fonction de la règle appliquée par le pré-processeur

- Règle1 ——————> Rien
- Règle2 ——————> Règle des K-plus proches voisins

(c) Choix des seuils s_1, s_2 *et* s_3
On choisit ces seuils de manière à ce que quand le pré-processeur ne peut pas transmettre un seul candidat alors il en élimine au moins 70% (on a remarqué expérimentalement que pour le vocabulaire des 10 chiffres le bon candidat se trouve toujours parmi les trois premiers candidats).

4. CALCUL DES DISTORSIONS PAR LE POST-PROCESSEUR

Pour effectuer ce calcul il y a trois possibilités, soit :

(a) Quantification de la forme de référence et de la forme à reconnaitre

(b) Quantification uniquement de la forme de référence par :

- Son propre ensemble de prototypes
- Tous les ensembles de prototypes
  (exemple : on peut exprimer la forme de référence "cinq", soit comme une suite de prototypes apppartenant tous à l'ensemble de prototypes cinq, soit appartenant à tous les ensembles de prototypes.)

(c) Pas de quantification

5. RESULTATS EXPERIMENTAUX

Pour évaluer les performances de cette dernière méthode nous avons utilisé la méthode fondée sur la quantification vectorielle multi-sections avec normalisation variable des mots pour le pré-processeur et la programmation dynamique pour le post-processeur.

Les tables 4.f, 4.m et 4.mf montrent les résultats de cette dernière méthode. Le post-processeur utilise des formes non quantifiées. Ces résultats montrent qu'il est possible d'obtenir un score final supérieur à celui obtenu par la méthode de programmation dynamique qu'on a cité dans ce chapitre et pour un nombre réduit de calculs. En effet, dans cette expérience le pré-processeur transmet au plus 3 candidats et par conséquent 3 calculs de score par programmation dynamique sont évalués au lieu de 10.

Locuteurs masculins		Locuteurs féminins	
dc	100	cf	95
pd	100	da	97.5
jp	95	ab	97.5
fl	100	cb	92.5
fa	100	nc	97.5
pn	100	bw	100
Total	99.16%		96.66%

Table 4.m | Table 4.f

Locuteurs mixtes			
dc	100	cf	95
pd	100	da	92.5
jp	95	ab	92.5
fl	97.5	cb	100
fa	100	nc	97.5
pn	100	bw	100
Total	98.75%		96.25%
Moyenne	97.5%		

Table 4.mf

La table 5 est la matrice de confusion de notre système de reconnaissance. On remarque que les chiffres 5 et 6 ont été confondu respectivement trois et cinq fois avec le chiffre 7. Ceci est peut-être dû aux confusions entre `/k/, /s/ et /t/`. Par contre le chiffre 7 n'a jamais été confondu avec aucun autre chiffre. Ces erreurs ne sont pas toutes commises par le post-processeur puisque le pré-processeur ne transmet pas parfois le bon candidat. En effet, sur 12 erreurs 3 sont dues à ce que le bon candidat n'a pas été transmis et 3 parce que le pré-processeur a transmis un seul candidat. Par contre le post-processeur a faussé 1 fois la décision correcte du pré-processeur.

	0	1	2	3	4	5	6	7	8	9
0	48	0	0	0	0	0	0	0	0	0
1	0	46	1	0	1	0	0	0	0	0
2	0	0	48	0	0	0	0	0	0	0
3	0	0	0	48	0	0	0	0	0	0
4	0	0	0	0	48	0	0	0	0	0
5	0	0	1	0	0	44	0	3	0	0
6	0	0	0	0	0	0	43	5	0	0
7	0	0	0	0	0	0	0	48	0	0
8	0	0	1	0	0	0	0	0	47	0
9	0	0	0	0	0	0	0	0	0	48

Table 5 : matrice de confusion

9.7 CONCLUSION

Nous avons présenté l'utilisation de la quantification vectorielle en reconnaissance automatique de mots. Cette technique permet d'obtenir des résultats de bonne qualité, avec une réduction des temps de calcul et de la place mémoire , notamment en reconnaissance multi-locuteurs, par rapport aux techniques classiques de type programmation dynamique. Un premier système décrit dans ce chapitre est fondé sur la quantification vectorielle multi-sections avec moyennage de toutes les trames de signal constituant une section. Nous avons ensuite proposé un système à deux étages, un pré-processeur fondé sur une quantification des formes et effectuant une première sélection rapide de candidats, et un post-processeur de comparaison fine par programmation dynamique. Ce dernier système a permis d'obtenir un taux de reconnaissance de 97.5% pour le vocabulaire des chiffres français multi-locuteurs.

10 RECONNAISSANCE DE LA PAROLE CONTINUE

10.1 INTRODUCTION

En reconnaissance de mots isolés il existe beaucoup de méthodes utilisant la quantification vectorielle. Pour pouvoir appliquer ces méthodes en reconnaissance de la parole continue nous avons choisi le phonème comme unité de reconnaissance. Comme le phonème est fortement influencé par son contexte gauche et droit, nous avons retenu pour chaque phonème uniquement sa partie spectralement stable (au sens d'une certaine mesure).

10.2 DESCRIPTION DE LA BASE DE DONNEES

Nous disposons d'une base de parole continue (corpus du texte lu "La bise et le soleil" du Greco Communication Parlée), dont nous avons extrait les locutions de six locuteurs masculins (BP, JB, FL, FC, GM et BG). Quatre locuteurs ont servi pour l'apprentissage du système de reconnaissance et les deux autres pour le test. Chaque locuteur a lu une fois le même texte qui dure en moyenne une quarantaine de secondes. Toutes les locutions ont été quantifiées sur 16 bits et échantillonnées avec une fréquence de 16 Khz.

10.3 APPRENTISSAGE

Chaque locution est étiquetée manuellement par un phonéticien (F. Lonchamp) en phonèmes (début et fin de chaque phonème dans le signal temporel). On découpe chaque phonème en trames de longueur fixe. D'après le chapitre précédent et [11,12], très peu de prototypes suffisent pour caractériser une section en reconnaissance mono-locuteur. Durant l'apprentissage on parcourt toutes les locutions et pour chaque phonème on calcule son centre de gravité, puis on retient toutes les trames qui se trouvent à une distance inférieure à un seuil S du centre de gravité. On obtient ainsi un nombre suffisant de trames pour chaque phonème. Par quantification vectorielle on garde un nombre limité de prototypes pour chaque phonème.

10.4 SEGMENTATION

La reconnaissance d'une phrase comprend d'abord la segmentation du signal en zones stables. Cette segmentation se fait selon deux principes différents : segmenter pour reconnaître et reconnaître pour segmenter. Pour faciliter ce travail on commence par identifier les phonèmes vocaliques.

1. Segmentation des phonèmes vocaliques.
   On calcule la courbe d'énergie dans la bande de fréquence [0..4000 Hz]. Après lissage on détecte tous les sommets de la courbe. Chaque sommet ainsi que la trame qui le précède et la trame qui le suit sont pris pour représentants de la zone à segmenter. On garde autour de ce sommet toutes les trames dont la distance par rapport à ces trois représentants n'excède pas un seuil donné.

2. Segmentation du reste de la phrase
   La segmentation du reste de la phrase se fait à son tour en deux étapes :

   - Segmentation par reconnaissance.
     On commence par comparer chaque trame du signal temporel aux représentants de tous les phonèmes non vocaliques. Chaque trame est identifiée au phonème qui réalise la plus petite distorsion. Après lissage, cette comparaison fait apparaître une succession de sous-suites ayant la même étiquette. Chaque sous-suite de longueur supérieure à un seuil donné est considérée comme une zone spectralement stable.
   - Segmentation du reste de la phrase
     La segmentation du reste de la phrase se fait de la manière suivante : toutes les zones de longueur (en trames) supérieure à un seuil donné (longueur minimale d'un phonème) sont considérées comme étant des zones segmentées. Pour la suite de ce chapitre on appelle ces zones des segments inconnus.

10.5 RECONNAISSANCE

La reconnaissance d'une phrase segmentée en zones stables (segments vocaliques, sous-suites et segments inconnus) se fait en trois phases :

- Reconnaissance des segments vocaliques

  Afin d'affiner la décision de la segmentation on augmente l'ordre de prédiction, puis on calcule de nouveau le centre de gravité de la zone en question. On prend le centre de gravité, la trame qui précède et la trame qui suit pour représentants de la zone. Seules les trames dont la distance par rapport à ces trois représentants ne dépasse pas un certain seuil sont retenues. En suite, on compare chaque trame de la nouvelle zone aux prototypes représentant chaque phonème vocalique. On garde pour chaque trame les dix candidats qui réalisent les dix premières plus petites distorsions. On fournit pour chaque zone les trois phonèmes candidats qui réalisent les trois plus petites distorsions moyennes.

- Reconnaissance des sous-suites

  La comparaison se fait de la même manière que précédemment, en ne considérant que les phonèmes non vocaliques.

- Reconnaissance des segments inconnus

  Lors des comparaisons on élimine tous les phonèmes vocaliques ainsi que certains phonèmes détectables avec une grande probabilité par la méthode des "sous-suites" comme `/j/, /ch/, /gh/ et /#/`.

10.6 RESULTATS

Nous utilisons le LPC pour la paramétrisation du signal vocal. L'ordre de LPC utilisé est 14 pour la recherche des zones stables et 20 pour la reconnaissance. La phrase à reconnaître est "La bise et le soleil se disputaient, chacun assurant qu'il était le plus fort, quand ils ont vu un voyageur qui s'avançait" (figure 5.9 représente l'étiquetage manuel de cette phrase pour les deux locuteurs servant de test). Nous avons utilisé 16 prototypes pour représenter chaque phonème. Pour la reconnaissance nous avons utilisé la distorsion d'Itakura-Saito à gain normalisé (d_{is}) et pour la recherche des zones stables nous avons utilisé d, une forme symétrique de cette distorsion :

$$d(x,y) = \frac{1}{2}(d_{is}(x,y) + d_{is}(y,x))$$

x et y étant deux vecteurs LPC normalisés par le gain.

SEGMENTATION

La phrase à identifier contient 33 phonèmes vocaliques. La segmentation automatique de la phrase prononcée par le locuteur FL a fait apparaître 33 phonèmes correctement segmentés en zones stables (le phonème `/an/` ($< 404, 415 >$) a été découpé en deux segments).
Pour le locuteur GM, 32 phonèmes ont été correctement segmentés, le phonème `/a/` ($< 791, 797 >$) a été détecté, mais fortement influencé par son contexte gauche. Le phonème `/z/` ($< 705, 709 >$) constitue la seule fausse détection de notre système. Le phonème `/i/` ($< 54, 67 >$) a été découpé en deux segments ($< 54, 58 >$ et $< 59, 67 >$), ceci est sans aucune importance pour le système de reconnaissance.
On remarque que, sans aucune connaissance préalable, pour beaucoup de segments le début ou la fin coïncident exactement avec l'étiquetage manuel du phonéticien.

RECONNAISSANCE MONO-LOCUTEUR

En plus des quatres locuteurs (BG, BP, JB et FC) les deux locuteurs (FL et GM) ont participé à l'élaboration du corpus d'apprentissage mais ils n'ont pas prononcé la phrase servant de test.

La figure 5.10 et la figure 5.11 représentent les résultats (mono-locuteur et multi-locuteurs) de notre système de reconnaissance pour les deux locuteurs servant de test.
La table 1 et la table 2 fournissent le résultat de la reconnaissance des voyelles pour les deux locuteurs. Chaque table fournit le pourcentage de reconnaissance quand le phonème à reconnaître est le premier candidat, puis l'un des deux premiers candidats et ensuite l'un des trois premiers candidats. On a porté sur ces tables uniquement les voyelles qui apparaissent dans la phrase servant de test.

On remarque que, pour le locuteur GM, seul le phonème `/un/` n'a pas été reconnu dans les trois premiers candidats, ce phonème a été confondu avec `/ai in a /`, donc le phonème `/un/` a été confondu avec `/in/` en deuxième position.

Pour le locuteur FL trois phonèmes n'ont pas été reconnus dans les trois premiers candidats. Ces phonèmes (marqués par * sur la dernière ligne) sont : /un/ qui a été confondu avec /in) a /, le phonème /)/ avec /ai a oe / et le phonème /&/ avec /ai e a /. Ce sont des phonèmes proches l'un de l'autre.

Pour le locuteur GM sur 22 sous-suites détectées il y a eu une seule fausse détection d'une voyelle, 20 correctement identifiées dans le premier choix et 21 correctement identifiées dans les deux premiers choix. Par contre, pour le locuteur FL, sur 21 sous-suites détectées 17 ont été correctement identifiées par le premier candidat, 18 correctement identifiées parmi les deux premiers candidats et 3 n'ont pas été identifiées parmi les trois premiers candidats.

	1	3	2	5	1	5	6	2	4	1	3	33
	on	an	un	i	e	ai	a	)	y	oe	&	pourcentage
1-er candidat	1	2	1	5	1	3	4	1	4	0	1	69.69%
2 candidats	1	3	1	5	1	5	6	1	4	0	2	87.87%
3 candidats	1	3	1	5	1	5	6	1	4	1	2	91%
			*					*			*	

Table 1 : Reconnaissance mono-locuteur des voyelles (locuteur FL)

	1	3	2	5	1	5	5	2	4	1	3	32
	on	an	un	i	e	ai	a	)	y	oe	&	pourcentage
1-er candidat	1	3	1	5	0	1	3	1	4	1	2	68.75%
2 candidats	1	3	1	5	0	5	5	1	4	1	3	90.62%
3 candidats	1	3	1	5	1	5	5	2	4	1	3	96.87%
			*									

Table 2 : Reconnaissance mono-locuteur des voyelles (locuteur GM)

RECONNAISSANCE MULTI-LOCUTEURS

Les phrases formant le corpus d'apprentissage ont été prononcées par les quatre locuteurs masculins (BP, BG, JB et FC). La phrase à reconnaître a été prononcée par deux autres locuteurs masculins (FL et GM). Les tables 3 et 4 fournissent les résultats de la reconnaissance multi-locuteurs pour les deux locuteurs FL et GM.

On remarque que, pour le locuteur FL le score final est le même que celui de la reconnaissance mono-locuteur. Les trois phonèmes non reconnus dans les trois premiers candidats sont : le phonème /)/ confondu avec /a ai in /, Le phonème /y/ confondu avec /i e un / et le phonème /&/ avec /ai e y /. Les deux derniers sont confondus avec des phonèmes spectralement voisins.

Pour le locuteur GM les résultats de la reconnaissance multi-locuteurs ont été nettement moins bons que ceux de la reconnaissance mono-locuteur. En particulier, sept phonèmes n'ont pas été reconnus dans les trois premiers candidats. Ces phonèmes sont : le phonème /on/ qui a été confondu avec /an) o /, le phonème /un/ avec /ai a e /, le phonème /e/ avec /eu y & /, le phonème /)/ avec /un o on /, le phonème /oe/ avec /eu & ai / et le phonème /y/ a été confondu une fois avec /e ai) / et une deuxième fois avec /i ai e /. Ces résultats montrent que le plus grand nombre de confusions sont faites entre phonèmes voisins.

Pour le locuteur FL, sur 21 sous-suites détectées il y a eu une seule fausse détection d'une voyelle, 2 sous-suites n'ont pas été identifiées parmi les trois premiers candidats, 16 ont été correctement identifiées en première position et 18 correctement identifiées parmi les deux premiers candidats. Sur les 19 sous-suites détectées, en reconnaissance multi-locuteurs du locuteur GM il y a eu deux fausses détections de voyelles, 14 ont été correctement identifiées en première position et 17 correctement identifiées parmi les deux premiers candidats.

	1	3	2	5	1	5	6	2	4	1	3	33
	on	an	un	i	e	ai	a	)	y	oe	&	pourcentage
1-er candidat	1	2	2	5	1	1	3	1	3	0	2	63.63%
2 candidats	1	3	2	5	1	5	5	1	3	1	2	87.87%
3 candidats	1	3	2	5	1	5	6	2	3	1	2	91%
								*	*		*	

Table 3 : Reconnaissance multi-locuteurs des voyelles (FL).

	1	3	2	5	1	5	5	2	4	1	3	32
	on	an	un	i	e	ai	a	)	y	oe	&	pourcentage
1-er candidat	0	3	1	5	0	4	0	1	1	0	2	53.12%
2 candidats	0	3	1	5	0	5	4	1	1	0	3	68.75%
3 candidats	0	3	1	5	0	5	5	1	2	0	3	87.12%
	*		*		*			*	*	*		

Table 4 : Reconnaissance multi-locuteurs des voyelles (GM).

10.7 CONCLUSION

Notre méthode fondée entièrement sur la quantification vectorielle permet d'obtenir un bon résultat en reconnaissance des phonèmes vocaliques et certains phonèmes comme /s/, /ch/, /z/, /j/ et /gh/. Les tables 5 et 6 résument les résultats de notre système de reconnaissance pour les voyelles. Pour la reconnaissance mono-locuteur nous obtenons sur les trois premiers candidats un score avoisinant les 94% et pour la reconnaissance multi-locuteurs nous obtenons un score final de 85.5%. En plus la plupart des erreurs commises sont dues à

des confusions entre phonèmes spectralement voisins.

Un large pourcentage des sous-suites détectées, aussi bien pour le locuteur FL que pour le locuteur GM, ont été correctement identifiées. Malgré l'absence totale de toute information provenant du pitch beaucoup de segments inconnus ont été aussi correctement identifiés.

	Locuteur FL	Locuteur GM	Total
1-er candidat	69.69%	68.75%	69.22%
2 candidats	87.87%	90.62%	89.24%
3 candidats	91%	96.87%	93.88%

Table 5 : Reconnaissance mono-locuteur des voyelles

	Locuteur FL	Locuteur GM	Total
1-er candidat	63.63%	53.12%	58.37%
2 candidats	87.87%	68.75%	78.30%
3 candidats	91%	78.12%	84.51%

Table 6 : Reconnaissance multi-locuteurs des voyelles


```
‹ 1    21 ›   /# /
‹ 21   29 ›   /l /
‹ 29   39 ›   /a /
‹ 39   52 ›   /b /
‹ 52   67 ›   /i /
‹ 67   81 ›   /z /
‹ 81   87 ›   /e /
‹ 87   91 ›   /l /
‹ 91  100 ›   /& /
‹ 100 116 ›   /s /
‹ 116 124 ›   /) /
‹ 124 127 ›   /l /
‹ 127 138 ›   /ai/
‹ 138 149 ›   /j /
‹ 149 165 ›   /s /
‹ 165 173 ›   /& /
‹ 173 179 ›   /d /
‹ 179 182 ›   /* /
‹ 182 189 ›   /i /
‹ 189 200 ›   /s /
‹ 200 207 ›   /p /
‹ 207 209 ›   /* /
‹ 209 218 ›   /y /
‹ 218 229 ›   /t /
‹ 229 233 ›   /* /
‹ 233 254 ›   /ai/
‹ 254 312 ›   /# /
‹ 312 326 ›   /ch/
‹ 326 337 ›   /a /
‹ 337 343 ›   /k /

‹ 343 348 ›   /* /
‹ 348 358 ›   /un/
‹ 358 370 ›   /a /
‹ 370 386 ›   /s /
‹ 386 395 ›   /y /
‹ 395 403 ›   /R /
‹ 403 416 ›   /an/
‹ 416 426 ›   /k /
‹ 426 429 ›   /* /
‹ 429 436 ›   /i /
‹ 436 448 ›   /l /
‹ 448 448 ›   /ai/
‹ 448 454 ›   /t /
‹ 454 461 ›   /* /
‹ 461 469 ›   /ai/
‹ 469 474 ›   /l /
‹ 474 483 ›   /k /
‹ 483 492 ›   /p /
‹ 492 495 ›   /* /
‹ 495 499 ›   /l /
‹ 499 507 ›   /y /
‹ 507 522 ›   /f /
‹ 522 544 ›   /) /
‹ 544 557 ›   /R /
‹ 557 570 ›   /e /
‹ 570 574 ›   /! /
‹ 574 601 ›   /e /
‹ 601 619 ›   /hh/
‹ 619 625 ›   /# /
‹ 625 632 ›   /k /

‹ 632 635 ›   /* /
‹ 635 644 ›   /an/
‹ 644 647 ›   /t /
‹ 647 654 ›   /* /
‹ 654 662 ›   /l /
‹ 662 669 ›   /l /
‹ 669 677 ›   /z /
‹ 677 694 ›   /on/
‹ 694 704 ›   /v /
‹ 704 718 ›   /y /
‹ 718 731 ›   /un/
‹ 731 738 ›   /v /
‹ 738 742 ›   /w /
‹ 742 749 ›   /a /
‹ 749 758 ›   /j /
‹ 758 768 ›   /e /
‹ 768 778 ›   /gh/
‹ 778 793 ›   /oe/
‹ 793 801 ›   /R /
‹ 801 808 ›   /k /
‹ 808 811 ›   /* /
‹ 811 817 ›   /l /
‹ 817 830 ›   /s /
‹ 830 841 ›   /a /
‹ 841 847 ›   /v /
‹ 847 860 ›   /an/
‹ 860 878 ›   /s /
‹ 878 902 ›   /ai/
‹ 902 952 ›   /# /
```

‹LABISA.FL›

```
‹ 1    20 ›   /# /
‹ 20   28 ›   /l /
‹ 28   38 ›   /a /
‹ 38   49 ›   /b /
‹ 49   50 ›   /* /
‹ 50   69 ›   /i /
‹ 69   82 ›   /z /
‹ 82   92 ›   /e /
‹ 92   97 ›   /l /
‹ 97  106 ›   /& /
‹ 106 121 ›   /s /
‹ 121 131 ›   /) /
‹ 131 137 ›   /l /
‹ 137 149 ›   /ai/
‹ 149 156 ›   /j /
‹ 156 175 ›   /s /
‹ 175 183 ›   /& /
‹ 183 191 ›   /d /
‹ 191 192 ›   /* /
‹ 192 200 ›   /i /
‹ 200 210 ›   /s /
‹ 210 216 ›   /p /
‹ 216 219 ›   /* /
‹ 219 228 ›   /y /
‹ 228 239 ›   /t /
‹ 239 243 ›   /* /
‹ 243 274 ›   /ai/
‹ 274 330 ›   /# /
‹ 330 345 ›   /ch/
‹ 345 356 ›   /a /

‹ 356 364 ›   /k /
‹ 364 368 ›   /* /
‹ 368 387 ›   /un/
‹ 387 397 ›   /a /
‹ 397 414 ›   /s /
‹ 414 424 ›   /y /
‹ 424 432 ›   /R /
‹ 432 447 ›   /an/
‹ 447 455 ›   /k /
‹ 455 461 ›   /* /
‹ 461 468 ›   /i /
‹ 468 472 ›   /l /
‹ 472 483 ›   /ai/
‹ 483 492 ›   /t /
‹ 492 496 ›   /* /
‹ 496 509 ›   /ai/
‹ 509 512 ›   /l /
‹ 512 521 ›   /& /
‹ 521 529 ›   /p /
‹ 529 532 ›   /* /
‹ 532 535 ›   /l /
‹ 535 543 ›   /y /
‹ 543 560 ›   /f /
‹ 560 581 ›   /) /
‹ 581 594 ›   /R /
‹ 594 660 ›   /# /
‹ 660 668 ›   /k /
‹ 668 673 ›   /* /
‹ 673 684 ›   /an/

‹ 684 690 ›   /t /
‹ 690 694 ›   /* /
‹ 694 700 ›   /i /
‹ 700 704 ›   /l /
‹ 704 715 ›   /z /
‹ 715 731 ›   /on/
‹ 731 742 ›   /v /
‹ 742 759 ›   /y /
‹ 759 765 ›   /? /
‹ 765 779 ›   /un/
‹ 779 786 ›   /v /
‹ 786 790 ›   /w /
‹ 790 794 ›   /a /
‹ 794 806 ›   /j /
‹ 806 818 ›   /a /
‹ 818 829 ›   /gh/
‹ 829 847 ›   /oe/
‹ 847 859 ›   /R /
‹ 859 865 ›   /k /
‹ 865 868 ›   /* /
‹ 868 874 ›   /i /
‹ 874 889 ›   /s /
‹ 889 899 ›   /a /
‹ 899 905 ›   /v /
‹ 905 918 ›   /an/
‹ 918 935 ›   /s /
‹ 935 960 ›   /ai/
‹ 960 965 ›   /hh/
‹ 965 986 ›   /# /
```

‹LABISA.GM›

/&/ : e muet /*/ : burst /hh/ : souffle /#/ : pause
/??/ : inconnu /!/ : bruit

Etiquetage manuel de la phrase : "La bise et le soleil se disputaient, chacun assurant qu'il était le plus fort, quand ils ont vu un voyageur qui s'avançait".

Figure 5.9: Etiquetage manuel de la phrase servant de test. Chaque phonème est repéré par son début et sa fin dans le signal temporel.

```
‹31    38›   /a in ai /   ==>a
‹53    67›   /i e y /     ==>i
‹82    87›   /e ai in /   ==>e
‹91    98›   /& ai a /    ==>&
‹118  122›   /ai a oe /   ==>) *
‹125  133›   /ai a e /    ==>ai
‹167  172›   /ai e a /    ==>& *
‹185  189›   /i e u /     ==>i
‹212  219›   /y eu i /    ==>y
‹235  251›   /ai a e /    ==>ai
‹328  334›   /a ai in /   ==>a
‹349  362›   /in ) a /    ==>un *
‹366  368›   /a in ai /   ==>a
‹391  398›   /y i e /     ==>y
‹404  411›   /an ) on /   ==>an
‹412  415›   /an ) on /   ==>an
‹432  436›   /i e y /     ==>i
‹441  447›   /e ai i /    ==>ai
‹462  469›   /e ai y /    ==>ai
‹475  478›   /ai & un /   ==>&
‹499  507›   /y e u /     ==>y
‹525  546›   /) an on /   ==>)
‹637  642›   /an ) on /   ==>an
‹657  664›   /i e y /     ==>i
‹680  686›   /on un o /   ==>on
‹708  713›   /y u & /     ==>y
‹719  729›   /un ) & /    ==>un
‹745  758›   /ai a ) /    ==>a
‹756  767›   /ai a e /    ==>a
‹780  791›   /ai a oe /   ==>oe
‹813  818›   /i & e /     ==>i
‹832  840›   /a an in /   ==>a
‹847  856›   /) an on /   ==>an
‹882  892›   /ai a in /   ==>ai
Reconnaissance mono-locuteur des voyelles

‹1    13›    /# k f /    ==>#
‹41   50›    /b d p /    ==>b
‹68   81›    /z s p /    ==>z
‹102  115›   /s p z /    ==>s
‹135  146›   /j g k /    ==>j
‹154  165›   /s z p /    ==>s
‹193  199›   /s z p /    ==>s
‹276  307›   /# f k /    ==>#
‹315  327›   /ch gh s /  ==>ch
‹374  386›   /s z p /    ==>s
‹508  521›   /f # p /    ==>f
‹549  556›   /d p b /    ==>R *
‹565  575›   /R # k /    ==>#
‹578  629›   /# f k /    ==>#
‹649  655›   /s z p /    ==>" *
‹655  571›   /z ch s /   ==>l *
‹769  779›   /gh ch z /  ==>gh
‹792  798›   /R k f /    ==>R
‹821  829›   /s z # /    ==>s
‹861  878›   /z z p /    ==>s
‹917  945›   /# f k /    ==>#
Reconnaissance mono-locuteur des sous-suites

‹16   28›    /l d t /    ==>l
‹175  182›   /t d p /    ==>d
‹202  209›   /p k t /    ==>p
‹222  232›   /f p k /    ==>t
‹254  273›   /k t v /    ==>#
‹337  346›   /k f p /    ==>k
‹418  429›   /t k p /    ==>k
‹452  459›   /p t f /    ==>"
‹481  496›   /b p d /    ==>p
‹689  705›   /v t p /    ==>v
‹732  742›   /v t p /    ==>v
‹801  810›   /p k f /    ==>k
‹895  914›   /g t d /    ==>#
Reconnaissance mono-locuteur du reste de la phrase
```

```
‹31    38›   /a un in /   ==>a
‹53    67›   /i e y /     ==>i
‹82    87›   /e in ai /   ==>e
‹91    98›   /& ai a /    ==>&
‹118  122›   /a ai in /   ==>) *
‹125  133›   /ai a in /   ==>ai
‹167  172›   /ai e y /    ==>& *
‹185  189›   /i e y /     ==>i
‹212  219›   /y i u /     ==>y
‹235  251›   /a ai e /    ==>ai
‹328  334›   /a ai in /   ==>a
‹349  362›   /un in a /   ==>un
‹366  368›   /un a in /   ==>a
‹391  398›   /i e un /    ==>y *
‹404  411›   /an ) on /   ==>an
‹412  415›   /an ) on /   ==>an
‹432  436›   /i e y /     ==>i
‹441  447›   /e ai i /    ==>ai
‹462  469›   /e ai y /    ==>ai
‹475  478›   /& ai un /   ==>&
‹499  507›   /y e u /     ==>y
‹525  546›   /) an a /    ==>)
‹637  642›   /an ) o /    ==>an
‹657  664›   /i e in /    ==>i
‹680  686›   /on un an /  ==>on
‹708  713›   /y in ai /   ==>y
‹719  729›   /un oe a /   ==>un
‹745  758›   /ai e a /    ==>a
‹756  767›   /ai a e /    ==>a
‹780  791›   /ai oe a /   ==>oe
‹813  818›   /i o & /     ==>i
‹832  840›   /a un in /   ==>a
‹847  858›   /) an o /    ==>an
‹882  892›   /a ai e /    ==>ai
Reconnaissance multi-locuteurs des voyelles

‹1    13›    /# k f /    ==>#
‹40   50›    /b d p /    ==>b
‹68   80›    /z s p /    ==>z
‹104  114›   /s z p /    ==>s
‹134  148›   /j g ui /   ==>j
‹154  165›   /s z p /    ==>s
‹194  200›   /s z p /    ==>s
‹258  271›   /# k f /    ==>#
‹276  307›   /# f k /    ==>#
‹315  326›   /ch gh s /  ==>ch
‹374  386›   /s z p /    ==>s
‹513  521›   /f # p /    ==>f
‹551  558›   /d p t /    ==>R *
‹577  607›   /# f k /    ==>#
‹649  655›   /s z p /    ==>" *
‹670  576›   /z s p /    ==>z
‹769  779›   /gh ch z /  ==>gh
‹819  827›   /z s p /    ==>s
‹865  876›   /z s p /    ==>s
‹893  899›   /g b m /    ==>ai **
‹917  945›   /# f k /    ==>#
Reconnaissance multi-locuteurs des sous-suites

‹16   28›    /t d l /    ==>l
‹175  182›   /p t d /    ==>d
‹203  209›   /p t k /    ==>p
‹222  232›   /f k p /    ==>t
‹337  346›   /f k p /    ==>k
‹418  429›   /t p k /    ==>k
‹450  459›   /p f t /    ==>"
‹481  496›   /b p d /    ==>p
‹561  574›   /k R t /    ==>#
‹610  634›   /f k v /    ==>#
‹689  705›   /t v p /    ==>v
‹732  742›   /v p t /    ==>v
‹794  810›   /k f p /    ==>k
‹902  914›   /t k f /    ==>#
Reconnaissance multi-locuteurs du reste de la phrase
```

Figure 5.10: Résultats du système de reconnaissance pour le locuteur **FL**.


```
‹30    37›   /a ai & /    ==>a
‹54    58›   /i y e /     ==>i
‹59    67›   /i e y /     ==>i
‹84    93›   /eu y e /    ==>e
‹97   105›   /& eu oe /   ==>&
‹122  128›   /e ai ) /    ==>]
‹139  150›   /e ai a /    ==>ai
‹177  182›   /& ai eu /   ==>&
‹191  197›   /i y e /     ==>i
‹223  227›   /y i e /     ==>y
‹246  265›   /e ai a /    ==>ai
‹346  353›   /a ai oe /   ==>a
‹369  382›   /ai in a /   ==>un *
‹386  395›   /in a un /   ==>a
‹416  423›   /y & ai /    ==>y
‹433  444›   /an on ) /   ==>an
‹462  472›   /i e y /     ==>i
‹473  480›   /e ai i /    ==>ai
‹497  510›   /e ai a /    ==>ai
‹514  518›   /oe & a /    ==>&
‹536  542›   /y eu e /    ==>y
‹565  582›   /) o on /    ==>)
‹675  680›   /an on ) /   ==>an
‹695  700›   /i e y /     ==>i
‹705  709›   /e y & /     ==>z **
‹717  735›   /on o an /   ==>on
‹743  761›   /y e eu /    ==>y
‹765  778›   /un in oe /  ==>un
‹791  797›   /e i & /     ==>/a j/
‹807  817›   /ai a oe /   ==>a
‹833  843›   /oe & ai /   ==>oe
‹871  875›   /i e y /     ==>i
‹892  899›   /a oe ai /   ==>a
‹904  917›   /an on ) /   ==>an
‹939  952›   /ai a on /   ==>ai
Reconnaissance mono-locuteur des voyelles

‹4    20›    /# p k /    ==>#
‹22   29›    /l n b /    ==>l
‹39   49›    /b d g /    ==>b
‹70   82›    /z t v /    ==>z
‹111  119›   /s p z /    ==>s
‹160  171›   /s p z /    ==>s
‹199  205›   /z s p /    ==>s
‹211  217›   /p t s /    ==>p
‹256  273›   /g d m /    ==>ai **
‹278  331›   /# k p /    ==>#
‹332  344›   /ch gh f /  ==>ch
‹356  364›   /k # p /    ==>k
‹401  412›   /s z f /    ==>s
‹424  432›   /R w v /    ==>R
‹544  555›   /f # k /    ==>f
‹595  662›   /# f k /    ==>#
‹798  806›   /j ui ch /  ==>j
‹820  832›   /gh ch f /  ==>gh
‹847  854›   /R m w /    ==>R
‹878  885›   /s z # /    ==>s
‹926  933›   /s f # /    ==>s
‹968 1000›   /# k f /    ==>#
Reconnaissance mono-locuteur des sous-suites

‹16   28›    /t d l /    ==>l
‹175  182›   /p t d /    ==>d
‹203  209›   /p t k /    ==>p
‹222  232›   /f k p /    ==>t
‹337  346›   /f k p /    ==>k
‹418  429›   /t p k /    ==>k
‹450  459›   /p f t /    ==>"
‹481  496›   /b p d /    ==>p
‹561  574›   /k R t /    ==>#
‹610  634›   /f k v /    ==>#
‹689  705›   /t v p /    ==>v
‹732  742›   /v p t /    ==>v
‹794  810›   /k f p /    ==>k
‹902  914›   /t k f /    ==>#
Reconnaissance mono-locuteur du reste de la phrase
```

```
‹30    37›   /ai a & /    ==>a
‹54    58›   /i e y /     ==>i
‹59    67›   /i e y /     ==>i
‹84    93›   /eu y & /    ==>e *
‹97   105›   /& eu e /    ==>&
‹122  128›   /un o on /   ==>) *
‹139  150›   /ai e a /    ==>ai
‹177  182›   /& ai e /    ==>&
‹191  197›   /i y e /     ==>i
‹223  227›   /i ai y /    ==>y
‹246  265›   /ai e a /    ==>ai
‹346  353›   /ai a in /   ==>a
‹369  382›   /ai a e /    ==>un *
‹386  395›   /un a in /   ==>a
‹416  423›   /e ai ) /    ==>y *
‹433  444›   /an ) o /    ==>an
‹462  472›   /i y e /     ==>i
‹473  480›   /ai e i /    ==>ai
‹497  510›   /e ai an /   ==>ai
‹514  518›   /on un & /   ==>&
‹536  542›   /i ai e /    ==>y *
‹565  582›   /) o an /    ==>)
‹675  680›   /an o ) /    ==>an
‹695  700›   /i e y /     ==>i
‹705  709›   /ai & i /    ==>z **
‹717  735›   /an ) o /    ==>on *
‹743  761›   /y eu e /    ==>y
‹765  778›   /un eu & /   ==>un
‹791  797›   /e i ai /    ==>/a j/
‹807  817›   /ai a e /    ==>a
‹833  843›   /eu & ai /   ==>oe *
‹871  875›   /i e y /     ==>i
‹892  899›   /eu ai a /   ==>a
‹904  917›   /an on ) /   ==>an
‹939  952›   /ai a e /    ==>ai
Reconnaissance multi-locuteurs des voyelles

‹4    20›    /# k p /    ==>#
‹22   29›    /l n b /    ==>l
‹40   49›    /b d g /    ==>b
‹71   81›    /z t p /    ==>z
‹162  171›   /s p z /    ==>s
‹183  189›   /b d g /    ==>d
‹198  205›   /z s t /    ==>s
‹211  218›   /p s t /    ==>p
‹266  272›   /g d b /    ==>ai **
‹278  331›   /# k f /    ==>#
‹332  344›   /ch gh f /  ==>ch
‹355  362›   /k v f /    ==>k
‹401  414›   /gh s ch /  ==>s
‹544  555›   /f p k /    ==>f
‹606  668›   /# f k /    ==>#
‹798  806›   /j d ui /   ==>j
‹878  885›   /s z f /    ==>s
‹954  962›   /g l d /    ==>ai **
‹974 1000›   /# f k /    ==>#
Reconnaissance multi-locuteurs des sous-suites

‹108  119›   /s p z /    ==>s
‹153  159›   /t k p /    ==>s
‹230  243›   /t p v /    ==>t
‹447  459›   /t k p /    ==>k
‹483  494›   /t p k /    ==>t
‹521  533›   /p t b /    ==>p
‹585  605›   /v b m /    ==>#
‹683  692›   /b n d /    ==>t
‹781  788›   /v t w /    ==>v
‹820  830›   /f k v /    ==>gh
‹846  868›   /p t k /    ==>R
‹920  936›   /f p k /    ==>s
‹955  971›   /f t v /    ==>#
Reconnaissance multi-locuteurs du reste de la phrase
```

Figure 5.11: Résultats du système de reconnaissance pour le locuteur GM.

Chapitre 6

CONCLUSIONS ET PERSPECTIVES

1 CONCLUSIONS

Dans le deuxième chapitre nous avons présenté les différentes techniques usuelles de paramétrisation du signal vocal. Nous avons justifié le choix de la paramétrisation par la méthode de prédiction linéaire LPC.

Dans le troisième chapitre nous avons surtout banalisé la technique de quantification vectorielle. Cette technique, introduite pour la première fois dans notre laboratoire, est largement utilisée par une grande majorité des chercheurs travaillant dans le domaine de la reconnaissance de la parole. Nous avons aussi présenté plusieurs techniques de codage : codage scalaire, codage vectoriel, codage avec distorsion et codage sans distorsion.

Dans le quatrième chapitre nous avons établi le lien existant entre la paramétrisation par LPC et la théorie du filtrage numérique. Nous avons aussi traité les différents problèmes liés à la transmission de la parole, surtout les problèmes de stabilité et le problème de recherche d'un bon compromis entre débit de transmission et confort d'écoute. Nous avons réalisé un synthétiseur à base de LPC et à débit variable.

Dans le cinquième chapitre nous avons présenté l'utilisation de la quantification vectorielle en reconnaissance automatique de la parole. Cette technique permet d'obtenir des résultats de bonne qualité, avec une réduction des temps de calcul et de la place mémoire notamment en reconnaissance multi-locuteurs, par rapport aux techniques classiques de type programmation dynamique.
Un premier système décrit dans ce chapitre est fondé sur la quantification vectorielle multi-sections avec moyennage de toutes les trames de signal constituant une section. Nous avons ensuite proposé un système à deux étages, un pré-processeur fondé sur une quantification des formes et effectuant une première sélection rapide de candidats, et un post-processeur

de comparaison fine par une deuxième méthode utilisant des formes non quantifiées. Ce dernier système a permis d'obtenir un taux de reconnaissance de 97.5% pour le vocabulaire des chiffres français multi-locuteurs.
Nous avons aussi proposé une méthode de reconnaissance de la parole continue. Cette méthode est fondée sur la quantification vectorielle. Les résultats obtenus surtout en segmentation et reconnaissance des voyelles sont très encourageants. L'atout majeur de cette méthode est la procédure de segmentation des voyelles. Cette procédure est complètement indépendante du contexte mono et multi-locuteurs. Les résultats expérimentaux montrent que, le début ou la fin de plusieurs segments coïncide avec l'étiquetage manuel du phonéticien.

2 PERSPECTIVES

Le codage vectoriel permet une grande réduction du débit de transmission ou de stockage, il permet en particulier d'atteindre des débits de transmission impossibles avec la quantification scalaire. L'inconvénient du codage vectoriel est que toutes les composantes d'un vecteur reçoivent le même code indépendamment de leur importance. En effet, si les vecteurs à coder représentent des coefficients de réflexion (k_i ; $1 \leq i \leq p$) alors k_1 *et* k_2 sont proches de 1 ou -1 alors que les autres sont plutôt proches de 0 et donc leur contribution est moins importante que celle de k_1 *et* k_2. Une solution mixte consiste à utiliser un codage scalaire adéquat pour au moins k_1 et un codage vectoriel pour $(k_2, ..., k_p)$. Cette solution augmente forcément le débit de transmission, mais elle peut améliorer la qualité de reconstitution du signal original.

Pour les applications telle que la transmission de la parole l'étape de quantification est suivie par une étape de synthèse. La méthode qui donne une meilleure synthèse moyennant un débit de 9600 bits/s est la méthode d'Atal. Pour le synthétiseur que nous avons réalisé les positions des excitations sont presque transmises sans aucun codage, un gain en débit et en confort d'écoute est possible par l'utilisation du codage combinatoire proposé dans [4]. D'autre part, en examinant de plus près le modèle d'excitation on constate que certaines impulsions sont séparées par des intervalles qui ont presque tous la même longueur. Une concrétisation de cette remarque serait la détection du pitch par synthèse de la parole. Si cette méthode s'avère concluante, elle sera rapide et précise car elle ne nécessite ni la quantification des paramètres du modèle ni l'utilisation d'un filtre perceptuel.

Nous avons présenté différentes méthodes de reconnaissance fondées sur la quantification vectorielle multi-sections. Nous avons surtout montré que la normalisation linéaire par mot donne des résultats meilleurs que ceux obtenus par normalisation linéaire de tous les mots du vocabulaire à une même longueur. Ces résultats peuvent encore être améliorés par optimisation du choix :

1. des références
2. de la longueur d'une section
3. de la longueur à laquelle sont normalisées toutes les références du même mot du vocabulaire.

4. du nombre de prototypes pour représenter une section.

En reconnaissance de la parole continue nous avons montré que la quantification vectorielle permet d'obtenir des résultats satisfaisants surtout en reconnaissance et segmentation des noyaux vocaliques. Le calcul seul des distorsions est insuffisant pour la reconnaissance des autres zones, il faut faire appel à d'autres sources d'information telles que :

- des connaissances linguistiques : à savoir que tel phonème ne peut être suivi que par tels phonèmes et précédé que par tels autres.
- des probabilités : statistiquement on peut, par apprentissage sur un corpus représentatif établir pour chaque phonème la liste de tous les phonèmes qui le précèdent et de tous les phonèmes qui le suivent avec leurs probabilités respectives.
- enfin une amélioration certaine de ce système est l'utilisation des procédures de segmentation des fricatives et des plosives du système APHODEX (système expert de décodage acoustico-phonétique, développé dans notre laboratoire).

Bibliographie

[1] J. P. Adoul. La quantification vectorielle des signaux : approche algébrique. *Ann. Telecommun.*, pages 158–177, 1986.

[2] B. S. Atal et J. R. Remde. A new model of lpc excitation for producing natural-sounding at low bit rates. *Proc. IEEE Int. Conf. ASSP*, pages 614–617, 1982.

[3] R. Bellman. *Dynamic Programming.* Princeton, Univ. Press, 1957.

[4] M. Berouti, H. Garten, P. Kabal, et P. Mermelstein. Efficient computation and encoding of the multipulse excitation for lpc. *Proc. ICASSP-84*, pages 10.1.1–10.1.4, 1984.

[5] D. K. Burton, J. E. Shore, et J. T. Buck. Isolated-word recognition using multisection vecor quantization codebook. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 837–849, aug 1985.

[6] A. Buzo, A. H. Gray, R. M. Gray, et J. D. Markel. Speech coding based upon vector quantization. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 562–574, oct 1980.

[7] F. Charpillet. Un système de reconnaissance de parole continue pour la saisie de textes lus. Thèse de Docteur d'université en Informatique, Université de Nancy I, 1985.

[8] D. Y. Cheng, A. Gersho, B. Ramamurthi, et Y. Shoham. Fast search algorithm for vector quantization and pattern matching. *Proc. ICASSP-84*, pages 9.11.1–9.11.4, 1984.

[9] S. B. Davis et P. Mermelstein. Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences. *IEEE Transaction on Acoustics, Speech, and Signal Processing*, pages 357–366, 1980.

[10] Recherche et Developpement dans les Industries de la Langue. J. P. Haton, G. Perennou, ed., Cours INRIA, Paris, 1987.

[11] A. Gourinda et J. P. Haton. Reconnaissance rapide de mots isolés par quantification vectorielle multi-sections. 16 ème JEP SFA, Hammamet, Tunisie, 1987.

[12] A. Gourinda et J. P. Haton. Utilisation de la quantification vectorielle multi-sections en reconnaissance de mots isolés. Revue d'Acoustique, 1988, à paraître.

[13] J. P. Haton et J. S. Liénard. La reconnaissance de la parole. La Recherche, No 99, 1979.

[14] J. G. Fritsch. Etude et simulation d'une famille de codeurs hybrides temporels offrant des débits de 6 a' 12 kbits/s pour des applications de qualité sub-téléphonique. Thèse de Docteur d'université en Informatique, Université de Nancy I, 1988.

[15] A. Gersho. On the structure of vector quantizers. *IEEE Trans. Inf. Theory*, pages 157–166, mar. 1982.

[16] A. Gersho et Y. Shoham. Hierarchical vector quantization of speech with dynamic codebook allocation. *Proc. ICASSP-84*, pages 10.9.1–10.9.4, 1984.

[17] A. H. Gray, R. M. Gray, et J. D. Markel. Comparaison of optimal quantizations of speech reflection coefficients. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 8–23, feb. 1977.

[18] A. H. Gray et J. D. Markel. Quantization and bit allocation in speech processing. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 459–473, dec. 1976.

[19] R. M. Gray. Vecor quantization. *IEEE ASSP Magazine*, pages 4–29, apr. 1984.

[20] H. A. Hawkins, D. M. Wilkes, M. A. Clements, et M. H. Hayes III. Perceptual weightings and optimal pulse positioning in multipulse lpc speech coding. *Proc. ICASSP-85*, pages 13.2.1–13.2.4, 1985.

[21] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the I.R.E*, pages 1098–1101, 1952.

[22] F. Itakura. Minimum production residual principle applied to speech recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 67–72, feb. 1975.

[23] N. S. Jayant. Digital coding of speech waveforms : pcm, dpcm, and dm quantizers. *Proceedings of the IEEE*, pages 611–632, may 1974.

[24] D. Jouvet. Reconnaissance de mots connectés independamment du locuteur par des méthodes statistiques. *Actes du 6 ème Congrès AFCET-INRIA Reconnaissance des Formes et Intelligence Artificielle*, pages 65–72, Antibes, nov. 1987.

[25] B.-H. Juang, D. Y. Wong, et A. H. Gray. Distorsion performance of vector quantization for lpc voice coding. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 294–303, apr 1982.

[26] M. Kunt. *Traitement Numérique des Signaux*. Dunod, 1984.

[27] L. F. Lamel, L. R. Rabiner, A. E. Rosenberg, et J. G. Wilpon. An improved endpoint detector for isolated word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 777–785, aug. 1981.

[28] Y. Linde, A. Buzo, et R. M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on communications*, pages 84–95, jan. 1980.

[29] J. Makhoul. Linear prediction - a tutorial review. *Proc. IEEE*, pages 561–580, April 1975.

[30] J. Makhoul. Spectral analysis of speech by linear prediction. *IEEE Transactions on Audio and Electroacoustics*, pages 140–148, jun. 1973.

[31] J. Makhoul, S. Roucos, et H. Gish. Vector quantization in speech coding. *Proceedings of the IEEE*, pages 1551–1586, nov. 1985.

[32] J. D. Markel et A. H. Gray. Implemantation and comparaison of two transformed reflection coefficient scalar quantization methods. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 575–583, oct. 1980.

[33] J. D. Markel et A. H. Gray. *Linear Prediction of Speech*. Springer-Verlag, 1976.

[34] J. Di Martino. Contribution à la reconnaissance globale de la parole : mots isolés et mots enchaînés. Thèse de Docteur Ingenieur en Informatique, Université de Nancy I, 1984.

[35] L. Miclet et M. Dabouz. Low bit rate transmission of speech by vector quantization of the spectrum. *Speech Communication*, pages 27–42, North-Holland, mar. 1987.

[36] C. Myers, L. R. Rabiner, et A. E. Rosenberg. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 623–635, dec. 1980.

[37] C. S. Myers et L. R. Rabiner. A leval building dynamic time warping algorithm for connected word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 284–297, apr. 1981.

[38] H. Ney. The use of a one-stage dynamic programming algorithm for connected word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 263–271, apr. 1984.

[39] N. Nocerino, F. K. Soong, L. R. Rabiner, et D. H. Klatt. Comparative study of several distorsion measures for speech recognition. *Proc. ICASSP-85*, pages 1.7.1–1.7.4, 1985.

[40] A. V. Oppenheim et R. W. Schaffer. *Digital Signal Processing*. Prentice-Hall, Inc, Englewood Cliffs, 1975.

[41] Kuk-Chin Pan, F. K. Soong, et L. R. Rabiner. A vector-quantization-based preprocessor for speaker-independent isolated word recognition. *IEEE Trans. Acoust. Speech, Signal processing*, pages 546–560, june 1985.

[42] L. R. Rabiner. On creating reference templates for speaker independent recognition of isolated words. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 34–42, feb. 1978.

[43] L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, et J. G. Wilpon. Speaker-independent recognition of isolated words using clustering techniques. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 336–349, aug. 1979.

[44] L. R. Rabiner, S. E. Levinson, et M. M. Sondhi. On the use of hidden markov models for speaker-independent recognition of isolated words from a medium-size vocabulary. *AT&T Bell Laboratories Technical Journal*, pages 627–642, apr. 1984.

[45] L. R. Rabiner, A. E. Rosenberg, et S. E. Levinson. Considerations in dynamic time warping algorithms for discrete word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 575–582, dec. 1978.

[46] L. R Rabiner et R. W Scahfer. *Digital Processing of Speech Signals.* Prentice-Hall, Inc., Englewood Cliffs, New Jersy, 1978.

[47] L. R. Rabiner, M. M. Sondhi, et S. E. Levinson. A vector quantizer incorporating both lpc shape and energy. *IEEE Int. Conf. Acoust., Speech, Signal Processing*, mar. 1984.

[48] L. R. Rabiner, J. G. Wilpon, A. M. Quinn, et S. G. Terrace. On the application of embedded digit training to speaker independent connected digit recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 272–279, apr. 1984.

[49] A. M. Rosie. *Theorie de l'Information et de la Communication.* Dunod, Paris, 1971.

[50] H. Sakoe. Two-level dp-matching–a dynamic programming-based pattern matching algorithm for connected word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 588–595, dec. 1979.

[51] H. Sakoe et S. Chiba. Dynamic progamming algorithm optimization for spoken word recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 43–49, feb. 1978.

[52] M. R. Sambur, A. E. Rosenberg, L. R. Rabiner, et C. A. McGonegal. On reducing the buzz in lpc synthesis. *J. Acoust. Soc. Am.*, pages 918–924, Bell Laboratories, Murray Hill, New Jersy 07974, mar. 1978.

[53] Y. Shoham et A. Gersho. Efficient codebook allocation for an arbitrary set of vector quantizers. *Proc. ICASSP-85*, pages 43.7.1–43.7.4, 1985.

[54] J. E. Shore et D. K. Burton. Discrete utterance speech recognition without time alignment. *IEEE Trans. Inf. Theory*, pages 473–491, jul 1983.

[55] T. E. Tremain. The government standard linear predictive coding algorithm : lpc-10. *Speech Technology*, pages 40–49, April 1982.

[56] R. Viswanathan et J. Makhoul. Quantization properties of transmission parameters in linear predictive systems. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 309–321, june 1975.

[57] J. G. Wilpon et L. R. Rabiner. A modified k-means clustering algorithm for use in isolated work recognition. *IEEE Trans. Acoust. Speech, Signal Processing*, pages 587–594, jun. 1985.

[58] D. Y.Wong, Biing-Hwang Juang, et A. H. Gray. An 800 bit/s vector quantization lpc vocoder. *IEEE Transaction on Acoustics, Speech, and Signal Processing*, pages 770–780, oct. 1982.

NOM DE L'ETUDIANT : GOURINDA Ahmed

NATURE DE LA THESE : DOCTORAT DE L'UNIVERSITE DE NANCY I en Informatique

VU, APPROUVE ET PERMIS D'IMPRIMER

NANCY, le 18 MARS 1988 462

LE PRESIDENT DE L'UNIVERSITE DE NANCY I

Résumé

Ce travail consiste à appliquer la quantification vectorielle aux divers domaines de la parole, en particulier :

- codage de la parole
- restitution ou synthèse de la parole
- reconnaissance de mots isolés
- reconnaissance de la parole continue

En codage, la quantification vectorielle permet de transmettre la parole à faible débit, en particulier elle fournit une solution satisfaisante au taux de transmission élevé que nécessite la méthode multi-impulsionnelle d'Atal. En reconnaissance automatique de mots, segmentation et identification des phonèmes elle permet d'obtenir des résultats de bonne qualité, avec une réduction des temps de calcul et de la place mémoire notamment en reconnaissance multi-locuteurs, par rapport aux techniques classiques de type programmation dynamique.

Mots clés :

Trame, fenêtre d'analyse
Quantificateur
Ensemble de prototypes
Quantification scalaire, vectorielle et sphérique
Filtre perceptuel
Programmation dynamique
Section
Quantification vectorielle multi-sections