UNIVERSITÉ DE NEUCHÂTEL
INSTITUT DE MICROTECHNIQUE

ANALYSE DE LA PAROLE: RECONNAISSANCE MULTILOCUTEUR DE MOTS ISOLÉS POUR LES SYSTÈMES MINIATURISÉS

THÈSE

PRÉSENTÉE À LA FACULTÉ DES SCIENCES
POUR OBTENIR LE GRADE DE DOCTEUR ÈS SCIENCES

PAR

Abdelatif Mokeddem

IMPRIMERIE CENTRALE NEUCHÂTEL
1985

IMPRIMATUR POUR LA THÈSE

Analyse de la parole: reconnaissance multilocuteur de mots isolés pour les systèmes miniaturisés

de M *onsieur Abdelatif Mokeddem*

UNIVERSITÉ DE NEUCHATEL

FACULTÉ DES SCIENCES

La Faculté des sciences de l'Université de Neuchâtel, sur le rapport des membres du jury,

Messieurs F. Pellandini, H. Hügli et M. Kunt (EPF-Lausanne)

autorise l'impression de la présente thèse.

Neuchâtel, le *13 août 1985*

Le doyen:

H. Beck

TABLE DES MATIERES

3. CALCUL DE LA DISTANCE ENTRE MOTS

4. RECONNAISSANCE MULTILOCUTEUR PAR RECALAGE FREQUENTIEL

5. ADAPTATION AU LOCUTEUR

6. ANALYSE FACTORIELLE APPLIQUEE AUX ECHANTILLONS MULTILOCUTEUR DE LA PAROLE

7. TECHNIQUES DE CLASSIFICATION AUTOMATIQUE APPLIQUEES A LA RECONNAISSANCE MULTILOCUTEUR

Introduction

La parole est un moyen naturel de communication entre les hommes. Les efforts entrepris depuis une trentaine d'années pour mettre au point des systèmes de reconnaissance automatique de la parole ont obtenu un succès limité et ont surtout montré la nature complexe de la communication parlée. L'auditeur humain utilise plusieurs niveaux de traitement pour la reconnaissance de la parole. Le premier niveau est la détermination des caractéristiques du signal de parole lui-même, c'est-à-dire l'analyse acoustique. Viennent ensuite les niveaux phonétique, lexical, syntaxique, sémantique, etc. C'est dire combien chez l'homme la reconnaissance et la compréhension de la parole sont intimement liées.

Avec l'arrivée des ordinateurs rapides vers les années 1960 et surtout depuis 1970, permettant en particulier le traitement en temps réel, de gros efforts ont été déployés. Le lancement du projet ARPA aux Etats-Unis en 1971 a été décisif en ce qui concerne l'évolution du traitement de la parole /31/.

Les avantages de la reconnaissance automatique de la parole sont tellement importants que l'on trouve déjà sur le marché des dispositifs d'utilisation limitée, mais néanmoins efficaces. Citons certaines applications qui ont déjà vu le jour : saisie vocale de données, aide aux handicapés, chambre d'hôpital avec possibilités de commandes vocales pour le malade, commande vocale de robot, commande vocale d'une montre portable, etc.

La mise au point d'un système de reconnaissance automatique de la parole rencontre deux types de difficulté : les variations intra-locuteur et les variations inter-locuteur. De telles variations sont dues à la complexité physique de l'appareil vocal qui présente une soixantaine de muscles pouvant participer à la phonation /49/.

Les variations intra-locuteur concerne un même locuteur, qui tout en prononçant un même mot peut parler de différentes façons d'une fois à l'autre. Ces variations ont pour causes : le soin apporté à l'articulation, la vitesse d'élocution, le niveau sonore, etc.

Les variations inter-locuteur sont dues aux dimensions et proportions de

l'appareil vocal variables selon l'âge, le sexe et la taille, aux manières de réaliser tel ou tel son liées à l'anatomie et aux habitudes articulatoires de chaque individu.

Les systèmes qui existent actuellement peuvent reconnaître un ensemble de mots isolés, un ensemble de mots enchainés et même de la parole continue, mais avec d'importantes limitations (vocabulaire limité, syntaxe rigide, locuteur consciencieux). Les techniques utilisées sont celles de la reconnaissance des formes /17/,/24/,/82/.

La plupart des systèmes existants utilisent l'une des deux méthodes d'analyse du signal de parole suivantes : 1) méthode fréquentielle : la transformée de Fourier à court terme du signal de parole est obtenue par un banc de filtres passe-bandes suivis de détecteurs d'enveloppe /22/; 2) méthode temporelle : le signal de parole est représenté par des coefficients de prédiction linéaire (LPC) calculés à des intervalles de temps réguliers /55/.
L'alignement temporel entre la forme inconnue et la forme de référence constitue le problème majeur dans un système de reconnaissance de mots isolés ou enchaînés. Pour le résoudre, la technique de programmation dynamique est appliquée avec succès /36/,/78/,/84/.

La plupart des systèmes existants sont monolocuteur, c'est-à-dire qu'ils reconnaissent "bien" la voix d'un seul locuteur.
Il y a pourtant des applications typiquement multilocuteur (par exemple : réservation ou renseignement par téléphone). C'est pourquoi différents travaux sont menés pour résoudre le problème de la reconnaissance multilocuteur. Les méthodes utilisées sont basées sur l'un des principes suivants /32/ : 1) définition, pour chaque mot, d'un ensemble de caractéristiques indépendantes du locuteur /79/; 2) adaptation au locuteur /9/,/26/,/38/,/51/; 3) caractérisation statistique de la variation interlocuteur par une technique de regroupement /23/,/67/,/74/,/75/.

Un point important qu'il faut prendre en considération lors de la mise au point d'un système est sa taille, en particulier ses besoins en mémoire. On peut classer les systèmes de reconnaissance automatique de la parole d'après leur taille comme suit : 1) système miniature avec une ou quelques puces; 2) système sur une ou plusieurs cartes à microprocesseur; 3) gros système.

Dans ce travail

Le but du présent travail est l'étude et la mise au point de systèmes miniaturisables de reconnaissance multilocuteur de mots isolés en temps réel.

a) compression de données

Les algorithmes développés pour extraire les caractéristiques d'une locution sont décrits comme une suite d'opérations à effectuer : compression temporelle, normalisation en amplitude par zone et quantification. Chaque opération peut être ajustée par l'intermédiaire d'un ou plusieurs paramètres. L'évaluation du taux d'erreur en fonction des divers paramètres dans un environnement monolocuteur et multilocuteur nous permet d'effectuer un choix optimal des paramètres selon le degré de miniaturisation souhaité.

b) reconnaissance multilocuteur

Nous avons abordé l'aspect multilocuteur selon plusieurs approches.

La première approche envisagée concerne le recalage fréquentiel. Il s'agit d'éliminer le plus possible les différences entre les locuteurs (hommes, femmes et enfants) par un recalage des motifs spectraux des locutions.

La deuxième approche testée est l'auto-adaptation du système de reconnaissance à un nouveau locuteur. L'adaptation s'opère par modification des références pendant l'utilisation.

La troisième approche, le point le plus important de ce travail, consiste dans la création de références multilocuteur. On y décrit de manière statistique les variations inter-locuteur : Il s'agit de trouver, pour l'ensemble des formes différentes (les diverses prononciations) d'un même mot telles qu'elles sont rencontrées dans une large population, un nombre réduit de références qui les décrivent au mieux.
La méthode suppose l'existence d'un certain nombre de classes de prononciations au sein d'une population.
La question qui se pose est de savoir s'il existe réellement des classes de prononciation au sein d'une population. L'application de l'analyse factorielle permet de répondre à cette question. Avec cette méthode, on peut représenter les prononciations d'un ou plusieurs mots par plusieurs locuteurs dans un espace de dimension réduite, ce qui permet d'analyser visuellement la distribution des prononciations.

Par la suite, nous développons et appliquons à la reconnaissance multilocuteur des mots isolés deux algorithmes de classification basés respectivement sur une fonction-critère. Il s'agit de l'algorithme d'échange basé sur une fonction-critère (AEC) et de l'agorithme séquentiel basé sur une fonction-critère (ASC). Une comparaison par rapport à d'autres algorithmes appliqués à la reconnaissance multilocuteur des mots isolés est effectuée.
D'autres aspects du problème, tels que le choix du représentant, l'élimination des isolés, le nombre variable de classes par mot sont également étudiés, et des solutions sont proposées.

Contenu

Dans le chapitre 0, nous commençons par modéliser le système de reconnaissance automatique de la parole et nous définissons les termes qui seront utilisés. Nous situerons également ce travail dans le domaine de la communication parlée homme-machine.

L'analyse par 'banc de filtres passe-bandes' correspond au calcul de la transformée de Fourier à court terme. Ceci est rappelé dans le chapitre 1.

Dans le chapitre 2 nous présentons les algorithmes de traitement numérique, dont le but consiste à standardiser et à dégager les caractéristiques essentielles du signal de parole. L'aspect compression de données est une tâche importante du traitement numérique dans la perspective de réalisation des systèmes miniaturisables.

Dans le chapitre 3, nous présentons et comparons les différentes méthodes de comparaison entre deux mots, mis sous leur forme matricielle.

Le chapitre 4 est consacré aux méthodes de recalage fréquentiel.

Le chapitre 5 aborde le problème de l'auto-adaptation du système de reconnaissance au locuteur.

Dans le chapitre 6, nous développons une méthode de représentation des échantillons multilocuteur. Elle est basée sur l'analyse factorielle. Elle permet de voir dans un espace familier la répartition des échantillons, et de mettre en évidence l'existence éventuelle de classes de locuteurs typiques au sein d'une population.

Dans le chapitre 7, nous étudions et appliquons divers algorithmes de classification automatique pour 'faire sortir' les classes de locuteurs au

sein d'une population, ainsi que leurs représentants. D'autres aspects du problème y sont également traités.

0. Modelisation

0.1 Domaine de la communication parlée homme-machine

Le domaine de la communication parlée homme-machine peut être subdivisé selon la figure 0.1.

Dans la communication parlée homme-machine, le système de synthèse est prévu pour répondre vocalement à l'homme : c'est la machine qui "parle".

La reconnaissance du locuteur consiste à déterminer ou à vérifier l'identité d'un locuteur.

Dans la reconnaissance automatique de la parole, la machine "écoute" et "comprend".

Nous avons divisé le domaine de la reconnaissance automatique de la parole (RAP) en quatre sous-domaines : la reconnaissance des mots isolés, la reconnaissance de mots enchaînés, la reconnaissance et compréhension de la parole continue avec un vocabulaire et une syntaxe limités et enfin, la reconnaissance et la compréhension du langage naturel. Seuls les trois premiers sous-domaines sont envisageables aujourd'hui. La reconnaissance et la compréhension du langage naturel par une machine est encore un rêve dans l'état actuel des connaissances /48/.

Il y a une grande dégradation des performances quand un système de RAP entrainé par un locuteur particulier est utilisé par d'autres locuteurs. Ainsi, nous avons divisé chacun des sous-domaines en partie monolocuteur et multilocuteur.

0.2 Approche

Il existe deux approches permettant d'aborder la reconnaissance de la parole : l'approche globale et l'approche analytique /31/.
Dans l'approche globale, l'unité de base est le mot : le mot est considéré comme une entité indivisible. Une petite phrase, de très courte durée, peut aussi être considérée comme un mot.
Dans l'approche analytique, on tente de détecter et d'identifier les

composantes élémentaires de la parole que sont les phonèmes.

L'approche analytique permet de traiter de gros vocabulaires, alors que dans l'approche globale on est limité à des vocabulaires de petite ou moyenne taille.

Mais la méthode analytique a un grand inconvénient : l'extrême variabilité du phonème en fonction du contexte (effets de la coarticulation). C'est pourquoi la méthode globale a été choisie dans ce travail.

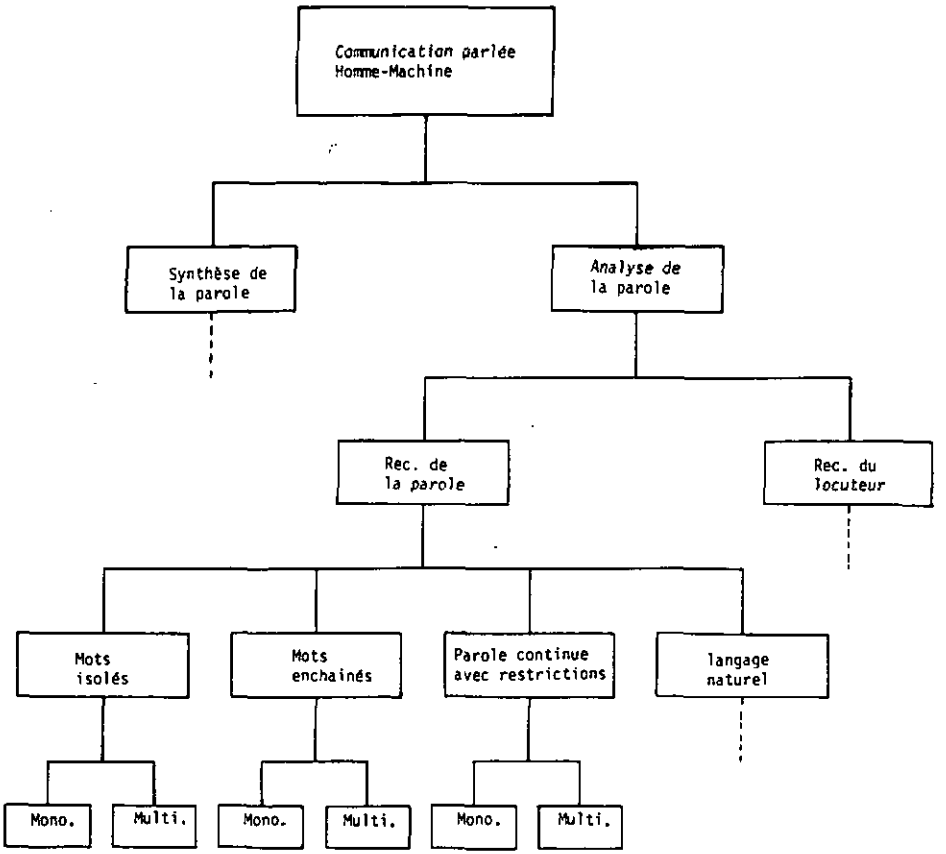


Fig.0.1 Domaine de la communication parlée homme-machine

0.3 Reconnaissance monolocuteur et multilocuteur

a) système de reconnaissance

Le principe utilisé est illustré dans la figure 0.2. Il comprend deux phases :

- la phase d'apprentissage : on crée pour chacun des mots une ou plusieurs formes de référence qui sont rangées dans un dictionnaire (mots de référence)

- la phase de reconnaissance : on identifie un mot inconnu à l'un des mots de référence rangés dans le dictionnaire.

Il y a deux opérations communes à la phase de reconnaissance et d'apprentissage : 1) analyse du signal acoustique, par exemple l'analyse spectrale par banc de filtres; 2) traitement numérique (détection des limites du mot, compression temporelle, normalisation en amplitude et quantification)

Le rôle du classifieur consiste à choisir le mot de référence le plus semblable au mot inconnu. Le choix est basé sur le calcul de la distance entre le mot inconnu et tous les mots de référence.

b) Création de références

Soit A_{mlv} la v-ième prononciation par le locuteur l du mot m et $\underline{A}_{mlv}$ le résultat obtenu après l'analyse spectrale et le traitement numérique. La création de référence consiste à déterminer, à partir d'un ensemble de prononciations $\{A_{mlv}\}$, un ensemble de références $\{\underline{R}_{mk}\}$. Considérons le cas de la création de références pour chaque mot de manière séparée. En supprimant l'indice m, la création de références devient :

$$\{A_{lv}\} \longrightarrow \{\underline{A}_{lv}\} \longrightarrow \{\underline{R}_{k}\}$$

c) Approches de reconnaissance multilocuteur

Dans ce travail, trois approches sont envisagées pour la reconnaissance multilocuteur :

1) recalage fréquentiel : on modifie le classifieur pour qu'il tienne

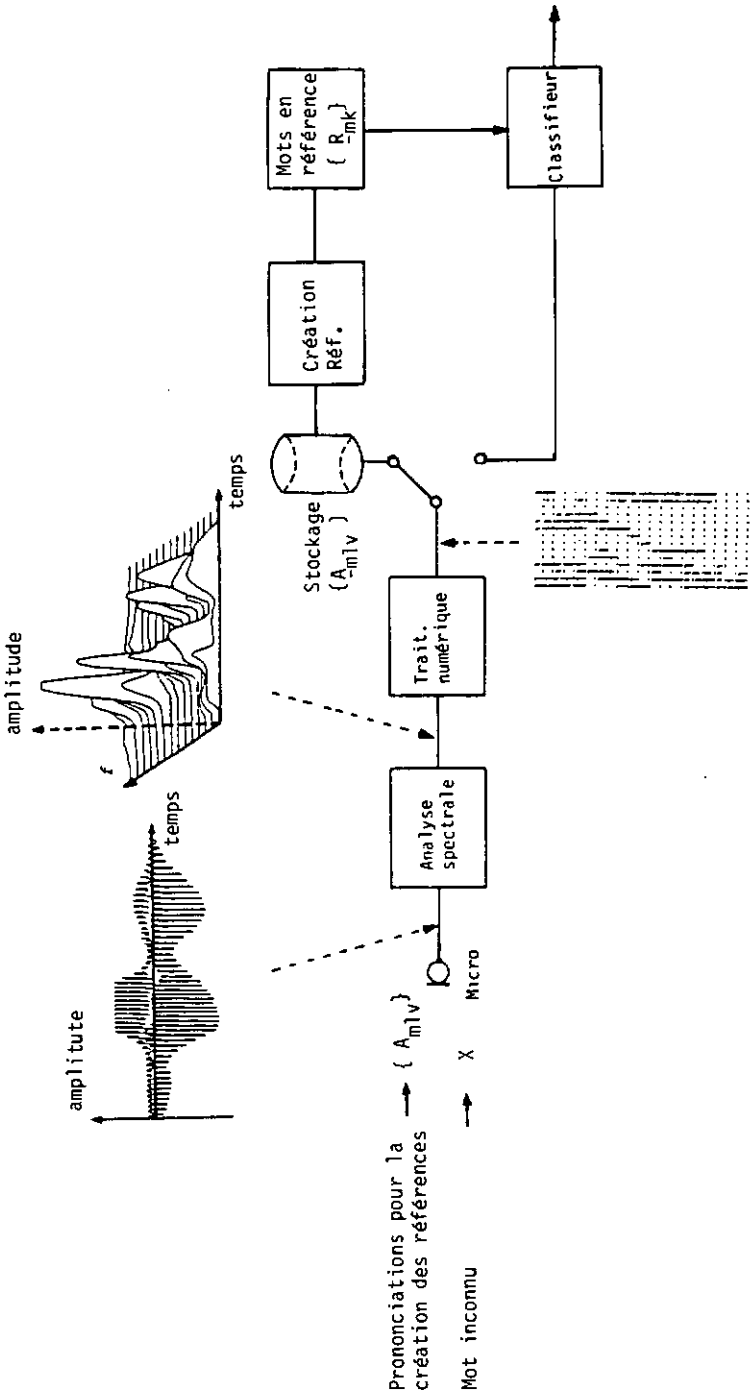


Fig.0.2 Principe de reconnaissance et d'apprentissage de la parole.

compte des variations inter-locuteur (chapitre 4).

2) Auto-adaptation au locuteur : un apprentissage initial est effectué sur un locuteur quelconque. Les références sont modifiées pendant la phase de reconnaissance pour être adaptées au nouveau locuteur (chapitre 5).

3) Apprentissage multilocuteur (Fig.0.2) : partant d'un grand nombre de prononciations A_{lv} pour chacun des mots, on détermine, d'après une technique de classification, un nombre limité de classes C_k, puis un représentant R_k pour chaque classe; les R_k constituent alors la référence multiple pour le mot donné (chapitre 7).

0.4 Taille du système de reconnaissance

Nous pouvons classer les systèmes de reconnaissance d'après leur taille comme suit :

1) systèmes miniatures :

il s'agit des systèmes intégrés sur une ou quelques puces possédant une taille mémoire de quelques Kbytes /9/.

2) systèmes moyens :

Il s'agit des systèmes réalisés sur une ou deux cartes à microprocesseur avec une taille mémoire d'environ 128 Kbytes.

3) gros systèmes (par exemple : DP100 de NEC, M1800 de VERBEX, etc.)

L'unité d'analyse spectrale utilisée dans ce travail (Fig.0.2) délivre 8400 bits pour un mot d'une durée moyenne de 500 ms. Il faut disposer donc d'une capacité mémoire d'environ 1 Kbyte par mot.

Comme notre travail va dans le sens de systèmes miniaturisables, il est nécessaire de réduire cette quantité de données. Nous montrons au chapitre 2 que celle-ci peut être fortement réduite, sans détérioration des performances. A titre d'exemple, dans la figure 0.2, après l'analyse spectrale (prétraitement analogique) et le traitement numérique, le nombre de bits par mot est ramené à 280 bits.

D.5 Décisions, taux d'erreur et taux de réjection

Soit $\underline{A}_m$ la forme finale de la prononciation du m-ème mot et $\underline{R}_{m'k'}$ la k'-ème forme de référence du m'-ième mot la plus proche de $\underline{A}_m$, c'est-à-dire :

$$D(\underline{A}_m,\underline{R}_{m'k'}) \leq D(\underline{A}_m,\underline{R}_{ij}) \quad \forall\ m'=1,\ldots,M \text{ et } k'=1,\ldots,K$$

où $D(\underline{A}_m,\underline{R}_{ij})$ est la distance entre $\underline{A}_m$ et $\underline{R}_{ij}$.

Il y a :

- réjection si la distance $D(\underline{A}_m,\underline{R}_{m'k'})$ dépasse un certain seuil.

- reconnaissance juste si $m = m'$ ($\underline{A}_m$ et $\underline{R}_{m'k'}$ représentent le même mot; la distance $D(\underline{A}_m,\underline{R}_{m'k'})$ étant inférieure au seuil de réjection

- erreur de reconnaissance si $m \neq m'$ ($\underline{A}_m$ et $\underline{R}_{m'k'}$ ne représentent pas le même mot; la distance $D(\underline{A}_m,\underline{R}_{m'k'})$ étant inférieure au seuil de réjection)

Soit Nt le nombre total de tests, Er le nombre total d'erreurs de reconnaissance et Rj le nombre total de réjections. On définit :

taux d'erreur = (Er/Nt).100%

taux de réjection = (Rj/Nt).100%

1. L'analyse spectrale à court-terme

1.1 Introduction

Beaucoup de systèmes de traitement de la parole sont basés sur le concept de l'analyse fréquentielle à court-terme qui est une représentation de la parole dans le domaine fréquentiel et temporel. Son succès est lié fondamentalement au fait que cette méthode d'analyse est proche de la nature du signal acoustique et de l'appareil auditif.

Le signal acoustique de la parole est un signal quasi-stationnaire. En conséquence, son spectre de fréquence à court-terme n'évolue que relativement lentement dans le temps. Ce spectre est donc une caractéristique propice à l'analyse de la parole puisqu'il contient l'essentiel de l'information acoustique et qu'il ne change que relativement lentement. L'analyse fréquentielle pratiquée exploite l'insensibilité de l'ouïe aux variations de phase et ne retient en conséquence que l'information relevante pour l'oreille, c'est-à-dire le spectre à court-terme du signal acoustique.

Le calcul de la transformée de Fourier complexe classique $X(w)$ de tout le signal $x(t)$ d'un mot serait inadéquat pour de multiples raisons. D'abord, ce calcul exige la connaissance de tout le signal acoustique et rend pratiquement impossible tout traitement en temps réel. Ensuite, comme c'est l'évolutivité de la parole dans le temps qui est recherchée, ni $X(w)$, ni $|X(w)|$ ne sont utiles; en effet, le spectre complexe $X(w)$ ne contient l'évolutivité que sous une forme implicite et quant au spectre d'amplitude $|X(w)|$, celle-ci manque complètement. La notion de transformée de Fourier à court-terme s'impose donc.

Dans ce chapitre, on montrera clairement le lien entre la définition théorique de la transformée de Fourier à court-terme (TFCT) et sa réalisation par un circuit pratique.

1.2 Transformée de Fourier à court-terme (TFCT)

D'une part, le signal x(t) est transformé de telle façon que l'intégration à un instant donné s'effectue seulement sur les valeurs du passé; d'autre part, pour mettre en évidence le phénomène d'évolutivité, on ne tient compte que du passé récent du signal x(t). On remplit ces exigences en multipliant le signal x(t) par une fenêtre temporelle h(t-t') (voir Fig.1.1)

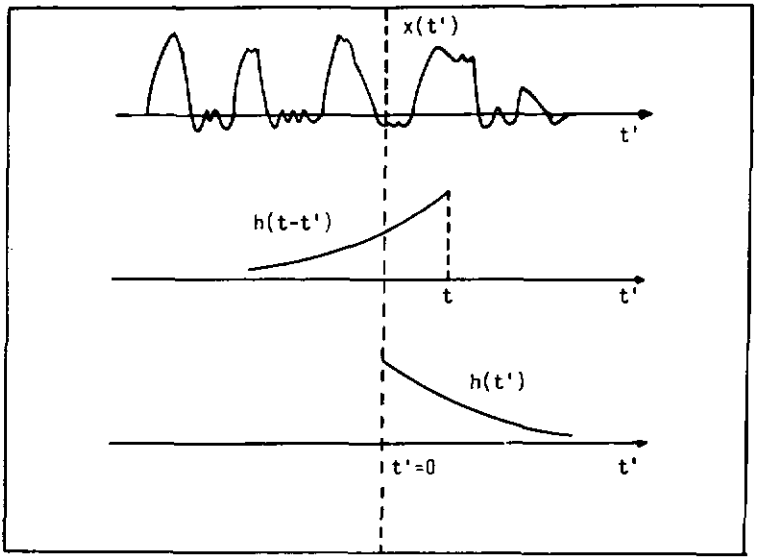


Fig.1.1 Illustration de la définition de la TFCT

Soit donc une fenêtre temporelle h(t-t') telle que :

$$h(t-t') = 0 \quad \text{si} \quad t' > t \qquad (1.1)$$

La définition de la transformée de Fourier à court-terme est la suivante:

$$X(w,t) = \int_{-\infty}^{\infty} x(t')h(t-t')\exp(-jwt')dt' \qquad (1.2)$$

On suppose naturellement que le choix de h(t-t') est fait de telle façon que cette intégrale existe.

On peut montrer (voir Flanagan /22/) que l'on peut retrouver le signal temporel x(t) à partir de la TFCT X(w,t). Ainsi, puisque la transformée inverse existe, l'information contenue dans le signal x(t) est entièrement véhiculée dans X(w,t).

1.3 Interprétation de la TFCT en termes de filtrage linéaire

La fenêtre temporelle h(t') peut être vue comme étant la réponse impulsionnelle d'un système physiquement réalisable, puisque h(t') = 0 pour t' < 0. Plus précisément, h(t') peut être interprétée comme étant la réponse impulsionnelle d'un filtre passe-bas.

En faisant un changement de variable t'' = t - t' dans la relation (1.2) il vient :

$$X(w,t) = \int_{\infty}^{0} x(t-t'')h(t'')\exp(-jw(t-t''))\ (-dt'')$$

$$= -\exp(-jwt) \int_{\infty}^{0} x(t-t'')h(t'')\exp(jwt'')\ dt''$$

$$= \exp(-jwt) \int_{0}^{\infty} x(t-t')h(t')\exp(jwt')\ dt' \qquad (1.3)$$

En introduisant le produit de convolution x(t)*h(t)exp(jwt), la relation (1.3) qui exprime la transformée de Fourier à court-terme peut être réécrite comme suit :

$$(1.3) \longrightarrow X(w,t) = \exp(-jwt)\ (x(t) * h(t)\exp(jwt)) \qquad (1.4)$$

à laquelle correspond le schéma fonctionnel de la figure 1.2.

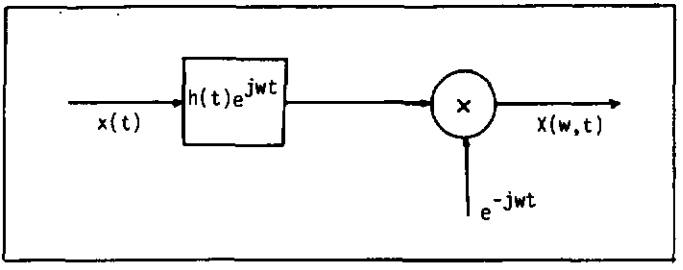


Fig.1.2 Schéma fonctionnel de la TFCT selon (1.4)

1.4 Mesure du module de la TFCT

Dans l'analyse de la parole, seul le module la TFCT nous intéresse. Naturellement, il peut être entièrement calculé si l'on définit la fenêtre h(t). Toutefois le temps de calcul est tellement important, qu'on ne peut envisager cette solution dans un système en temps réel. La détermination du module de la TFCT à l'aide d'un système physique tel qu'il est décrit dans ce qui suit est plus avantageuse.

D'après la figure 1.2 ou d'après la relation (1.4) on a :

$$|X(w,t)| = |\exp(-jwt)|.|x(t) * h(t)\exp(jwt)|$$

$$= |x(t) * h(t)\exp(jwt)| \qquad (1.5)$$

car

$$|\exp(-jwt)| = 1.$$

Le schéma fonctionnel en notation complexe pour la mesure du module de la TFCT a en conséquence la forme simple de la figure 1.3.

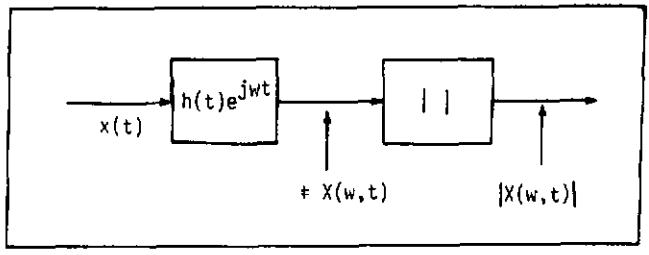


Fig. 1.3 Schéma fonctionnel pour la mesure du module de la TFCT

La réalisation physique correspondante apparait en développant (1.4) :

$$X(w,t) = \exp(-jwt)\left(\int_0^\infty x(t-t')h(t')\cos wt'\,dt' - \int_0^\infty x(t-t')h(t')\sin wt'\,dt'\right)$$

$$= \exp(-jwt)\,(a(w,t) + j.b(w,t)) \qquad (1.6)$$

Il vient :

$$|X(w,t)| = (a^2(w,t) + b^2(w,t))^{1/2} \qquad (1.7)$$

avec :

$$a(w,t) = x(t) * h(t)\cos(wt) \qquad (1.8)$$

$$b(w,t) = x(t) * h(t)\sin(wt) \qquad (1.9)$$

Ce qui conduit à la méthode de mesure du spectre d'amplitude de la figure 1.4.

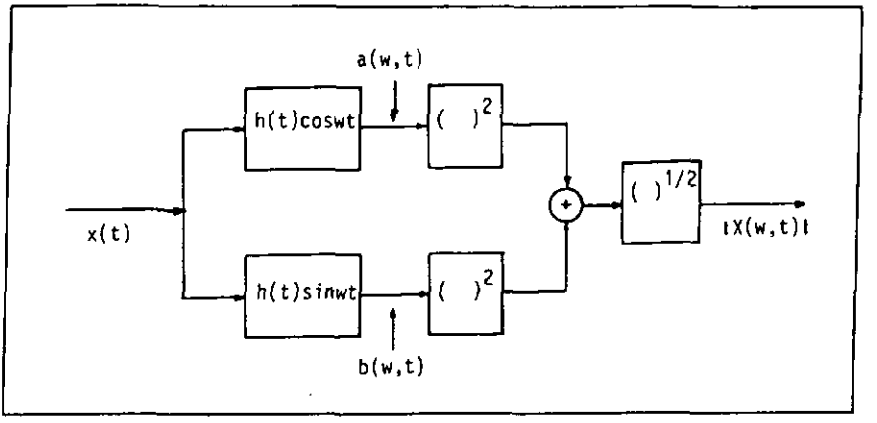


Fig.1.4 Réalisation physique du module de la TFCT

La réalisation physique selon la figure 1.4 conduit à un circuit relativement lourd. En introduisant la définition du détecteur d'enveloppe, *on peut la simplifier.*

On va montrer /22/ que, dans certaines conditions, l'amplitude $(a^2(w,t) + b^2(w,t))^{1/2}$ peut s'obtenir, à l'aide d'un détecteur d'enveloppe, à partir de a(w,t) tout seul ou de b(w,t) tout seul.

La transformée de Hilbert $\hat{x}(t)$ d'un signal x(t) est définie par le produit de convolution suivant :

$$\hat{x}(t) = x(t) * \frac{1}{\pi t} \qquad (1.10)$$

On va montrer que b(w,t) est la transformée de Hilbert de a(w,t). En utilisant la propriété d'associativité du produit de convolution, on a

$$\hat{a}(w,t) = H(\, x(t) * h(t)\cos wt\,) = x(t) * H(\, h(t)\cos wt)\,)$$

Si les spectres de h(t) et coswt ne se chevauchent pas /6/, /22/

$$\hat{a}(w,t) = x(t) * h(t)\, H(\, \cos wt\,) = x(t) * h(t)\sin wt$$

$$= b(w,t) \qquad (1.11)$$

Ainsi :

$$(a^2(w,t) + b^2(w,t))^{1/2} = (a^2(w,t) + \hat{a}^2(w,t))^{1/2} \qquad (1.12)$$

Donc, le module de la transformée de Fourier à court-terme X(w,t) s'obtient à partir de a(w,t) seul, en déterminant l'enveloppe de a(w,t).

1.5 Principe de réalisation pour l'analyse de la parole

En faisant varier w dans la figure 1.4, on obtient un banc de filtres passe-bande; chacun des filtres passe-bande est suivi par un détecteur d'enveloppe.

Ainsi, la mesure du module de de la TFCT conduit au schéma pratique d'analyse de la parole de la figure 1.5. Le signal électrique obtenu à la sortie du microphone (après avoir été amplifié et préaccentué) passe à travers un banc de filtres passe-bande. Le signal à la sortie de ces filtres passent ensuite à travers un circuit détecteur d'enveloppe qui peut être approché soit par un redresseur suivi d'un passe-bas, soit par un circuit RMS.
On obtient, ainsi, une approximation du module de la TFCT à court terme. Ces signaux analogiques sont convertis en signaux numériques et transmis.

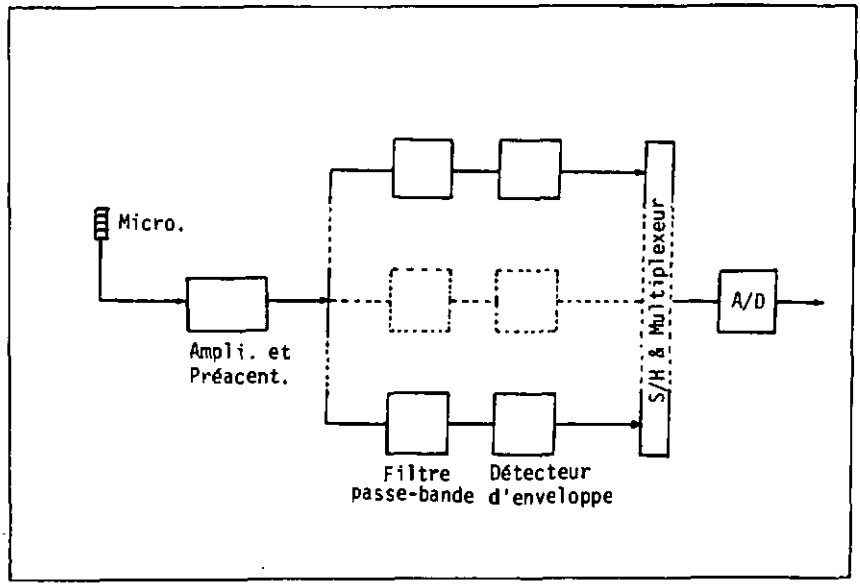


Fig.1.5 Schéma de principe du processeur analogique

1.6 Choix

En plus des considérations des paragraphes précédents, le traitement analogique exploite le modèle simplifié de l'appareil auditif. Ce dernier est supposé sensible au logarithme de l'énergie dans une bande fréquentielle, la taille de ces bandes étant proportionnelle à leur fréquence centrale.

Par ailleurs, l'appareil vocal ayant une certaine inertie, on a un phénomène de quasi-stationnarité. C'est pour cela qu'on se contente de 100 échantillons par seconde et par canal.

Le traitement analogique choisi (figure 1.6) est décrit dans ce qui suit.

Le signal à la sortie du microphone est préamplifié et passe ensuite à travers un filtre de préaccentuation, dont le rôle est de renforcer l'énergie des hautes fréquences. Ce filtre a un réponse en amplitude constante jusqu'à 0.33 kHz, une pente positive de 6dB/octave de 0.33 kHz à 4.8 kHz, une amplitude constante de 4.8 à 10 kHz, et une pente négative de 6dB/octave au delà de 10 kHz.

Ensuite, le signal passe à travers une série de filtres passe-bas de deuxième ordre (filtres d''antialising'), suivie d'un banc de quatorze filtres passe-bande. Les filtres passe-bande sont des filtres à capacité commutée. Leurs largeurs de bande sont réparties logarithmiquement selon l'axe fréquentiel (voir table ci-dessous)
Le choix des filtres à capacité commutée a été motivé par la faible surface d'intégration qu'ils exigent par rapport à d'autres types de filtres.

Les signaux de sortie des filtres passe-bande passent ensuite à travers les circuits RMS.

Les quatorzes signaux obtenus sont appliqués séquentiellement à un convertisseur A/D de 12 bits. Chaque canal est échantilloné à une fréquence de 100 Hz (globalement 1400 échantillons de la TFCT à la seconde). Le processeur analogique livre donc 100x14x12 = 16800 bits/seconde.

No du filtre	Fréquences à 3 dB en Hz
1	75 - 150
2	150 - 300
3	300 - 378
4	378 - 475
5	475 - 600
6	600 - 756
7	756 - 952
8	952 - 1200
9	1200 - 1512
10	1512 - 1905
11	1905 - 2400
12	2400 - 3024
13	3024 - 3810
14	3810 - 4800

Table 1.1 Fréquences à 3 dB des filtres passe-bandes

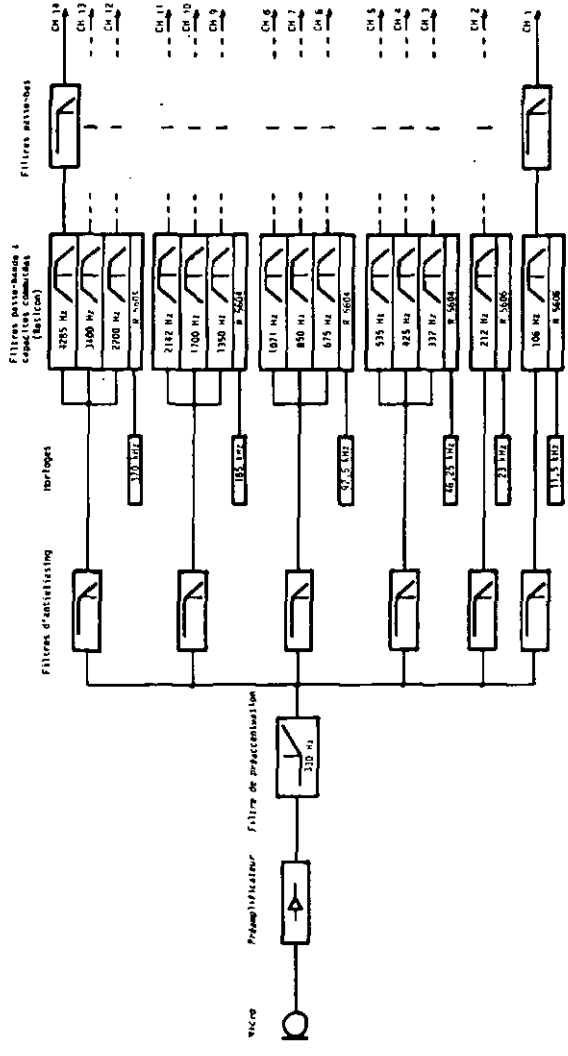


Fig.1.6 Détails de l'analyseur spectral à court-terme

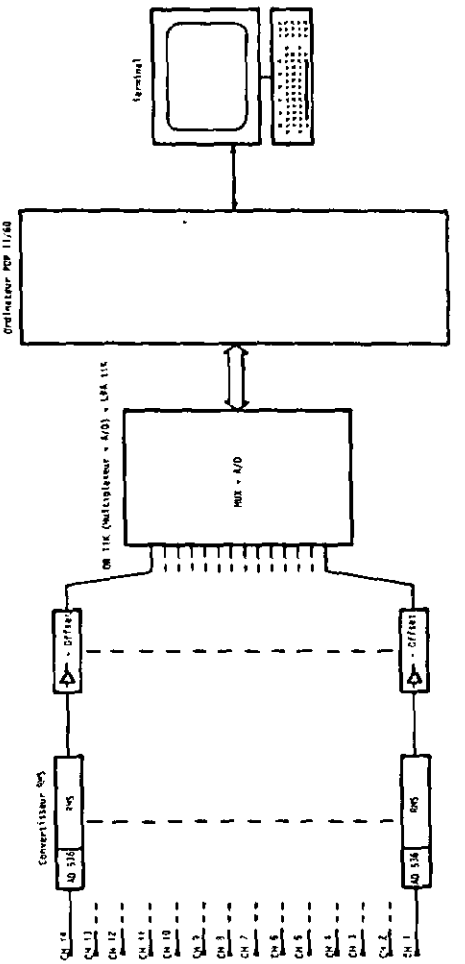

Terminal
Ordinateur PDP 11/60
MUX + A/D
Offset
Offset
Convertisseur RMS
RMS
RMS
AD 536
AD 536
CH 14
CH 13
CH 12
CH 11
CH 10
CH 9
CH 8
CH 7
CH 6
CH 5
CH 4
CH 3
CH 2
CH 1

Fig.1.6 (Suite)

2. Le traitement numérique

Le traitement numérique tel qu'il est défini ici comprend les étapes successives suivantes :

1) la détection des limites d'un mot

2) la compression temporelle

3) la normalisation d'amplitude

4) la quantification.

L'objectif de la compression temporelle, de la normalisation d'amplitude et de la quantification est de transformer les données relatives à un mot à la sortie de l'analyseur spectral en un petit ensemble de caractéristiques, par exemple sous la forme d'une matrice décrivant encore fidèlement les propriétés utiles à la reconnaissance du mot.
Chacune de ces opérations peut être ajustée par l'intermédiaire d'un ou de plusieurs paramètres. L'évaluation du taux d'erreur en fonction des divers paramètres dans un environnement monolocuteur et multilocuteur nous permet d'effectuer un choix optimal des paramètres selon le degré de miniaturisation souhaité.

2.1 Détection des limites d'un mot

Il existe un grand nombre de variantes d'algorithmes de détection des limites d'un mot /42/, /71/.

L'algorithme utilisé ici est décrit par les trois points suivants :

1. Nous avons un début de mot dès que l'amplitude moyenne

$$\bar{a}_m = \frac{1}{K} \sum_{k=1}^{K} a_m(k)$$

est supérieure à un certain seuil s; $a_m(k)$ est le signal à la sortie du

k-ième canal, échantilloné à l'instant m.

2. Nous avons une fin de mot, si on a déjà détecté un début de mot, et si l'amplitude moyenne $\bar{a}_m$ reste inférieure au seuil s pendant une durée supérieure à T_{sil} (typiquement 200ms).

3. La durée d'un mot doît être au moins égale à T_{min} (typiquement 200ms) et au plus égale à T_{max} (typiquement 2000ms).

Le paramètre T_{min} permet de rejeter un bruit pour autant qu'il dure moins que T_{min}. Le paramètre T_{sil} permet d'avoir des silences au sein d'un mot, si leurs durées sont inférieures à T_{sil} (par exemple, silence avant les plosives).

Adaptation du seuil au bruit ambiant

Le seuil s doit être adapté au bruit ambiant. Pendant une certaine durée (typiquement $10.T_e$, où T_e est la période d'échantillonnage par canal ; T_e = 10 ms), on mesure la moyenne du niveau de l'amplitude du signal du bruit ambiant :

$$s_{m_0} = \frac{1}{m_0} \sum_{m=1}^{m_0} \bar{a}_m$$

où m_0 est choisi égal à 10 (cela correspond à 100 ms). Le seuil est ensuite déterminé à l'aide de la relation de récurrence suivante :

$$s_m = \alpha . s_{m-1} + (1 - \alpha) . \bar{a}_m$$

ce qui correspond à un filtrage numérique passe-bas dont la fonction de transfert en z est :

$$H(z) = \frac{1 - \alpha}{1 - \alpha . z^{-1}}$$

Pratiquement, α est choisi égal à 0.9, ce qui correspond à un temps de montée d'environ $10.T_e$. En outre, on fixe la valeur minimale de s_m à une valeur s_{min} :

$$s_m = \text{Max}\ (s_m, s_{min})$$

Nous avons alors un début de mot dès que :

$$\bar{a}_m > (1+\beta)s_m \qquad \text{avec } \beta = 0.5$$

Jusqu'à la fin du mot, le seuil reste celui pour lequel cette dernière inégalité a été vérifiée pour la première fois.

L'organigramme de la figure 2.2 décrit complètement cet algorithme.

Cas d'une préaccentuation insuffisante

Dans le cas d'une préaccentuation insuffisante des hautes fréquences (leur niveau d'amplitude à la sortie du microphone est beaucoup plus petit que celui des basses fréquences), on effectue un traitement séparé des canaux basses fréquences et hautes fréquences. Dans ce cas, il faut déterminer deux seuils : $s1_m$ pour les canaux basses fréquences et $s2_m$ pour les canaux hautes fréquences. La détermination des seuils $s1_m$ et $s2_m$ est effectuée de la même façon que dans le cas où il n'y a pas de séparation de traitement entre les basses fréquences et les hautes fréquences.
Soit $\bar{a}1_m$ et $\bar{a}2_m$, respectivement la moyenne dans les canaux basses fréquences et hautes fréquences.

Les conditions de début de mot et de fin de mot deviennent

1. Nous avons un début de mot dès que :

$$\bar{a}1_m > s1_m \qquad \underline{ou} \qquad \bar{a}2_m > s2_m$$

2. Nous avons une fin de mot si on a déjà détecté un début de mot, et si les deux relations suivantes sont vérifées pendant T_{sil}

$$\bar{a}1_m < s1_m \qquad \underline{et} \qquad \bar{a}2_m < s2_m$$

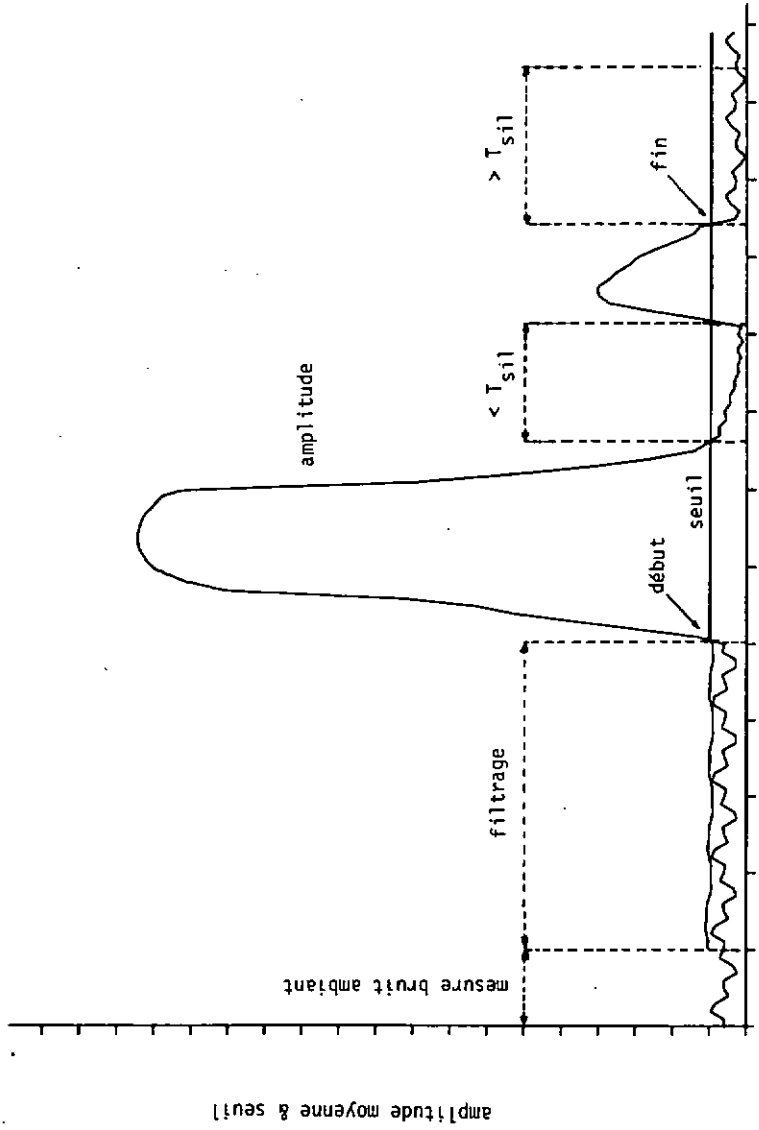


Fig.2.1 Détection de début et de fin de mot

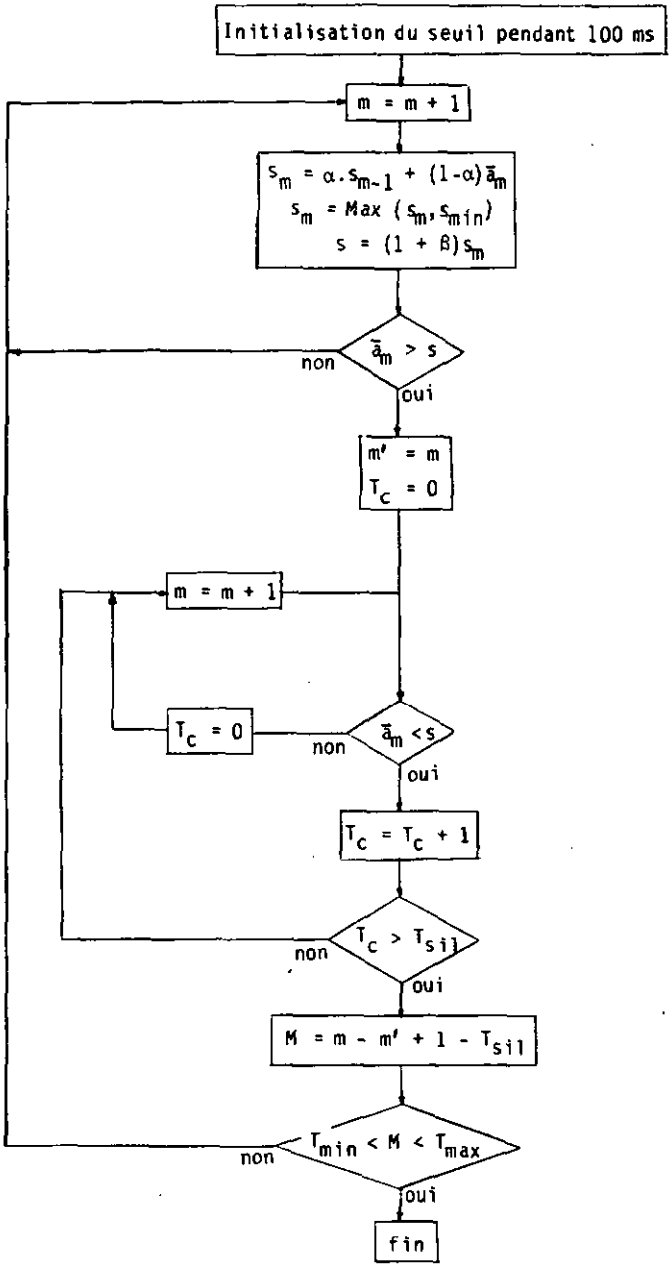


Fig.2.2 Algorithme de détection de début et de fin de mot

2.2 Normalisation temporelle

La durée des mots prononcés étant en général variable, les matrices brutes $\underline{A}(K,M) = \{a_m(k),\ k = 1, \ldots, K\ ;\ m = 1, \ldots, M\}$ auront un nombre variable d'échantillons temporels. Si le nombre des canaux K est fixe d'une matrice à l'autre, la durée temporelle M est variable. La normalisation temporelle, telle que nous l'entendons dans ce paragraphe consiste dans la transformation de la matrice $\underline{A}(K,M)$ où M est variable d'une prononciation à l'autre, en une une nouvelle matrice $\underline{B}(K,N)$ où N est fixe et généralement plus petit que M. Deux types de transformation sont étudiés ici :

1) la normalisation linéaire

2) la normalisation curviligne.

La normalisation linéaire a pour but de :

- standardiser et comprimer les mots en durée.

La normalisation curviligne /20/, /28/, /60/, /87/ a un second but :

- réduire systématiquement la durée des zones stables, mettant ainsi en évidence les zones de transition.

Lorsque la comparaison, ou calcul de distance, entre deux matrices est effectuée de façon linéaire, une normalisation temporelle est nécessaire. Par contre, elle n'est pas nécessaire dans le cas d'une comparaison dynamique (Chapitre 3).

2.2.1 Normalisation linéaire à longueur fixe

Elle consiste à comprimer linéairement la durée M d'un mot en une durée N fixe et généralement plus petite que M. Soit :

$\underline{a}_m$ $\quad m = 1, \ldots, M$ $\quad$ une suite temporelle de M vecteurs à normaliser
$\underline{b}_n$ $\quad n = 1, \ldots, N$ $\quad$ une suite temporelle de N vecteurs désirés

Les conditions aux limites sont fixées ainsi :

$$\underline{b}_1 = \underline{a}_1 \quad , \quad \underline{b}_N = \underline{a}_M$$

A l'indice n du vecteur $\underline{b}_n$ correspond un indice réel :

$$\alpha = (n-1)(M-1)/(N-1) + 1$$

Par la suite, différentes variantes peuvent être envisagées. Nous avons opté pour la variante du voisin le plus proche :

Soit m' l'entier le plus proche de α. Nous avons alors :

$$\underline{b}_n = \underline{a}_{m'}$$

Ceci revient à rendre variable le rythme d'échantillonnage des canaux.

2.2.2 Normalisation curviligne

Le signal de parole peut être considéré comme une fonction du temps m dans l'espace E^K (K est le nombre de canaux) :

$$W(m) = (a_m(1), a_m(2), \ldots, a_m(K))$$

Il y aura une faible variation de W(m) lorsqu'il y a des sons stables (généralement les voyelles, silence avant les plosives), et une forte variation lors des transitions entre des sons stables. La normalisation curviligne se base sur le fait que les transitions dans un mot sont plus caractéristiques du mot que les états stationnaires.

Soit $d_{W(m)}$ la distance entre deux valeurs successives de W(m), c'est-à-dire entre les deux vecteurs-fréquences successifs $\underline{a}_m$ et $\underline{a}_{m-1}$

$$d_{W(m)} = d(\underline{a}_m, \underline{a}_{m-1})$$

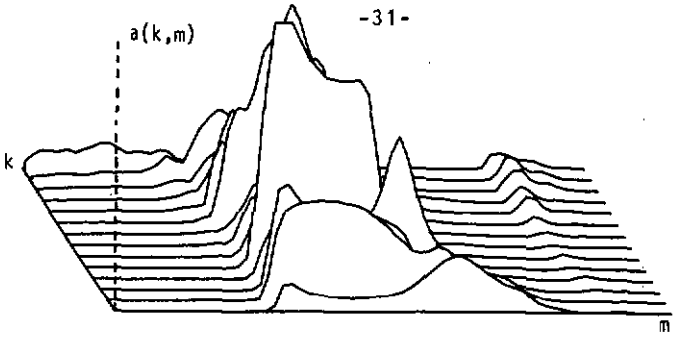
et $D_{W(m)}$ la distance cumulée à l'instant m

$$D_{W(m)} = \sum_{i=1}^{m} d_{W(i)}$$

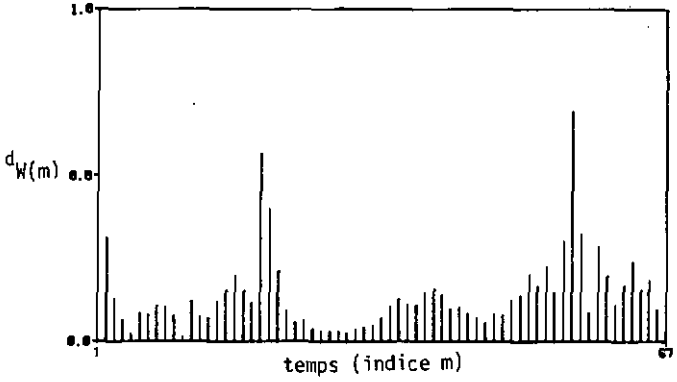
La valeur finale $D_{W(M)}$ représente la variation totale de la fonction W(m). La normalisation curviligne peut alors se faire selon l'illustration de la figure 2.3.
Soit N le nombre de vecteurs-fréquences à déterminer. On place sur l'axe $D_{W(m)}$ (Fig.2.3c) N points espacés uniformément avec un pas $\Delta D = D_{W(M)}/N$. A chacun de ces points sur l'axe $D_{W(m)}$ est ensuite associé un point m' sur

l'axe m. Au point m' nous associons alors le vecteur $\underline{a}_m$ dont l'indice m est le plus proche de m'.

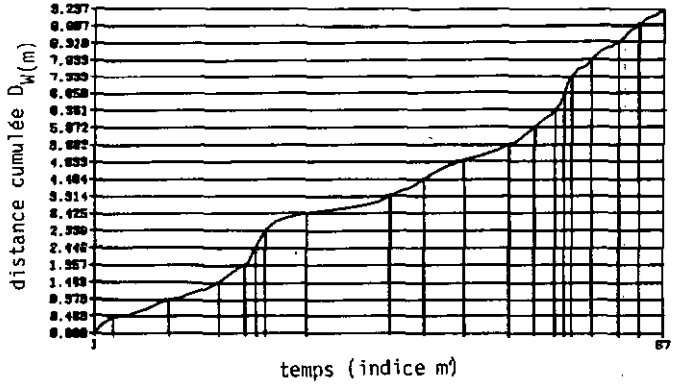
Diverses variantes peuvent être alors envisagées pour la définition du vecteur $\underline{b}_n$. Dans l'organigramme de la Fig.2.4, c'est la variante du successeur qui est indiquée : $\underline{b}_n = \underline{a}_m$, $\underline{a}_m$ étant le vecteur pour lequel $D_{W(m)}$ dépasse $(n-1)\,\Delta D$.



(a)



(b)



(c)

Fig.2.3. Illustration de la normalisation temporelle curviligne

(a) spectrogramme du mot "cinq"

(b) distance entre vecteurs-fréquences successifs $d_{W(m)}$

(c) distance cumulée $D_{W(m)}$ échantillonnée avec un pas de $D_{W(M)}/20$.

Algorithme

$\underline{a}_m$, m = 1,...,M les vecteurs d'entrée

$\underline{b}_n$, n = 1,...,N les vecteurs de sortie

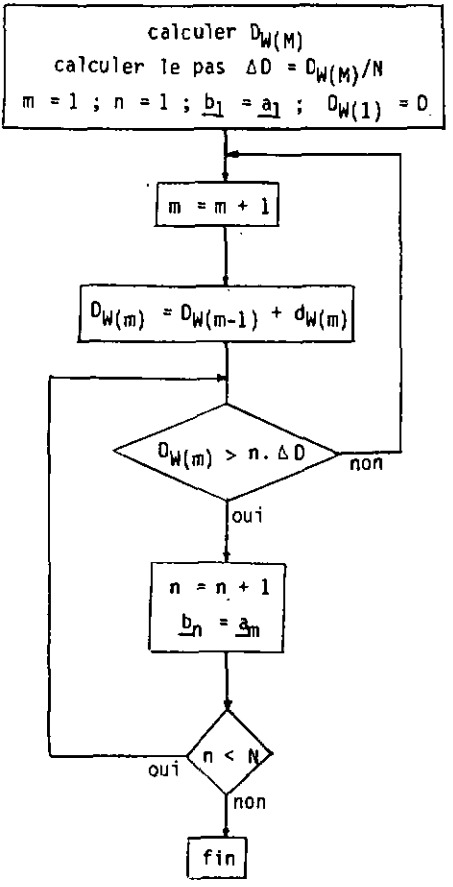


Fig.2.4 Algorithme de normalisation curviligne

2.3 Normalisation en amplitude

En général, le niveau d'amplitude d'un même mot varie d'une fois à l'autre. Il y a plusieurs raisons à cela : voix plus ou moins haute, distance locuteur-microphone plus ou moins longue, gain de l'amplification plus ou moins grand, etc. Pour compenser ces variations, il convient de normaliser en amplitude les éléments de la matrice brute.

La normalisation en amplitude consiste à diviser chaque élément a(k,m) de la matrice brute $\underline{A}$ par une valeur de normalisation f(k,m,$\underline{A}$) :

$$a'(k,m) = a(k,m) \;/\; f(k,m,\underline{A})$$

f(k,m,$\underline{A}$) dépend des éléments de la matrice brute $\underline{A}$. Nous allons définir plusieurs méthodes pour choisir f(k,m,$\underline{A}$). Ces méthodes seront testées plus loin (§2.5) en reconnaissance monolocuteur et en reconnaissance multilocuteur.

2.3.1 Normalisation en amplitude globale

Ici, la valeur de normalisation est la valeur moyenne de tous les éléments a(k,m) de la matrice.

$$f(k,m,\underline{A}) = \Big(\sum_{k=1}^{K} \sum_{m=1}^{M} a(k,m) \Big) \;/\; K.M$$

2.3.2 Normalisation en amplitude par zone

Dans certains cas, la normalisation en amplitude du paragraphe précédent est insuffisante. Prenons par exemple un même mot, prononcé de deux façons différentes, une fois en accentuant le début du mot et l'autre, en accentuant la fin. Une normalisation en amplitude selon le paragraphe précédent serait insuffisante, car elle ne tient pas compte des variations d'accentuation en fonction du temps, intervenant lors d'une prononciation d'un mot.

C'est pourquoi nous utilisons une normalisation en amplitude qui tient compte des variations globales et locales d'un mot d'une fois à l'autre. Ainsi, nous définissons la normalisation en amplitude par zone comme suit :

à chaque élément $a(k,m)$ de la matrice $\underline{A}$ est associée une zone (ou voisinage) Z (Fig.2.5). La matrice $\underline{Z}$ correspondant à une zone est une sous-matrice de $\underline{A}$:

$$\underline{Z} = \{ a(k,m) \;/\; k = k1,\ldots,k2 \;,\; m = m1,\ldots,m2 \}$$

La normalisation est effectuée ainsi : à chaque élément $a(k,m)$ de la matrice brute correspond un nouvel élément $a'(k,m)$ tel que :

$$a'(k,m) = a(k,l) \;/\; f(k,m,\underline{Z})$$

avec

$$f(k,m,\underline{Z}) = \Big(\sum_{k=k1}^{k2} \sum_{m=m1}^{m2} a(k,m) \Big) \;/\; (k2-k1+1).(m2-m1+1)$$

La quantité $f(k,m,\underline{Z})$ est la valeur moyenne de la zone Z. Si $f(k,m,\underline{Z}) = 0$ (c'est le cas où tous les éléments de $\underline{Z}$ sont nuls), nous posons $a'(k,m) = 0$.

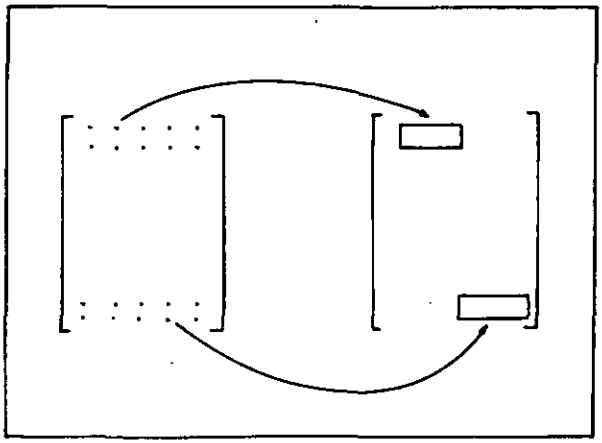


Fig.2.5 Illustration de la normalisation en amplitude par zone

Exemples de zones

a) toute la matrice (Z0) : c'est le cas de la normalisation en amplitude globale (§2.3.1)

b) ligne (Z1) :

$$a(k,m) \longrightarrow \{ \underline{Z1} = a(k,m) \ / \ m = 1,\ldots,M \}$$

La valeur de normalisation $f(k,m,\underline{Z1})$ pour un k donné, représente la valeur moyenne de l'amplitude de la TFCT obtenue sur le k-ième canal (Remarquons que $f(k,m,\underline{Z1})$ ne varie pas lorsque l'indice temporel m varie).

c) demi-colonne (Z2) :

$$a(k,m) \longrightarrow \underline{Z2} = \begin{cases} \{a(k,m) \ / \ k = 1,\ldots,K/2 \ \} & \text{si } 1 < k \leq K/2 \\ \{a(k,m) \ / \ k = K/2+1,\ldots,K \} & \text{si } K/2 < k \leq K \end{cases}$$

d) colonne (Z3) :

$$a(k,m) \longrightarrow \underline{Z3} = \{ a(k,m) \ / \ k = 1,\ldots,K \}$$

Les valeurs de normalisation $f(k,m,\underline{Z3})$, $m = 1,\ldots,M$ représentent les valeurs moyennes des amplitudes de la TFCT obtenues à la sortie des K canaux à l'instant m.

e) fenêtre rectangulaire de centre a(k,m), de longueur LF selon l'axe fréquentiel, et de longueur LT selon l'axe temporel (Z4) :

$$a(k,m) \longrightarrow \underline{Z4} = \{ a(k,m) \ / \ k = k1,\ldots,k2 \ , \ m = m1,\ldots,m2 \}$$

Les relations qui lient (k1, k2, m1, m2) à (k, m, LF , LT) sont :

$$k1 = \text{Max} (1 , k-LF/2)$$
$$k2 = \text{Min} (k+LF/2 , K)$$
$$m1 = \text{Max} (1 , m-LT/2)$$
$$m2 = \text{Min} (m+LT/2 , M)$$

Ces relations définissent la fenêtre rectangulaire à l'intérieur et au bord de la matrice.

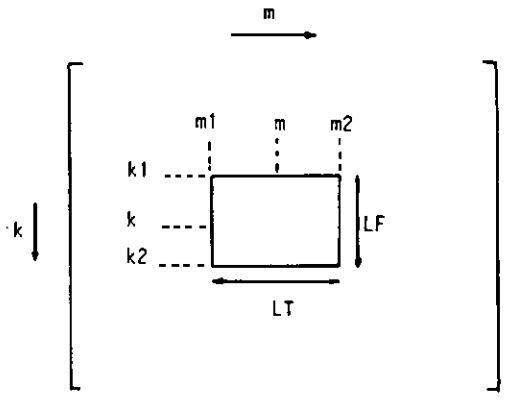


Fig.2.6

Limites de Z4

- si LF > 2K et LT > 2M , on retrouve la normalisation en amplitude globale (c'est Z0)

- si LF = 1 et LT > 2M , on normalise d'après la ligne (c'est Z1)

- si LF > 2K et LT = 1 , on normalise d'après la colonne (c'est Z3).

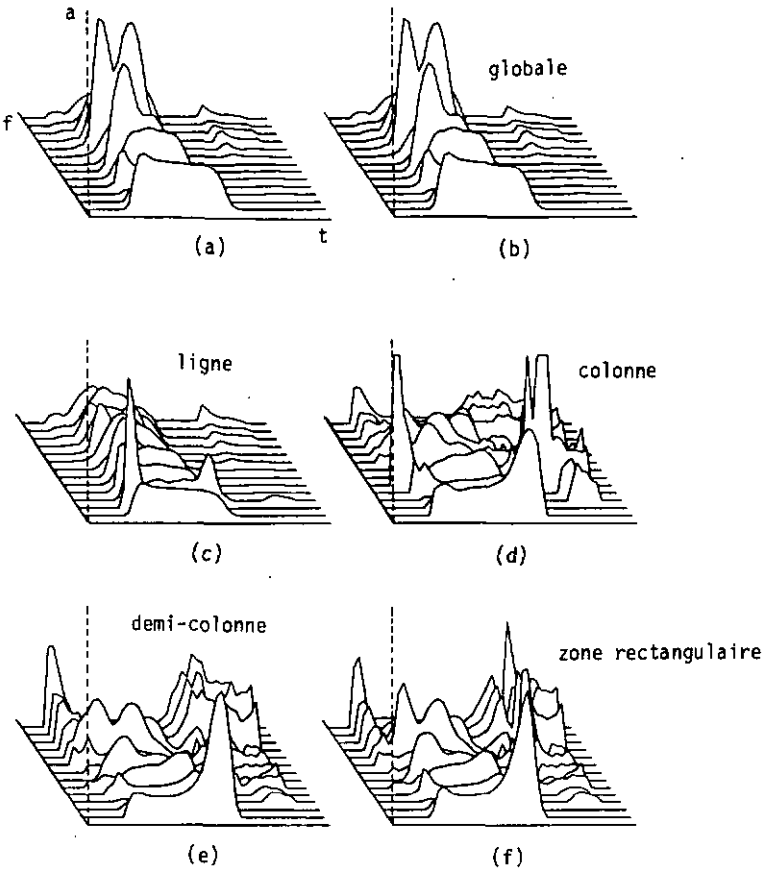


Fig.2.7 Illustration de la normalisation en amplitude

(a) spectrogramme du mot 'cinq' prononcé par une femme

(b) après Z0

(c) " Z1

(d) " Z2

(e) " Z3

(f) " Z4 (LF=28,LT=5)

2.4 Quantification

Le but de la quantification est de réduire la quantité de données sans détériorer les performances de reconnaissance. Elle revient à effectuer un choix optimal de paramètres qui serviront à définir la fonction de quantification. La fonction de quantification (linéaire ou logarithmique) peut être exprimée ainsi : à chaque élément a(k,m) (noté ci-après a) de la matrice A est associé un nouvel élément a'(k,m).
Dans notre cas, l'élément a(k,m) est l'élément de la matrice obtenue après les normalisations temporelle et en amplitude.

$$q = f(a) \qquad 0 < q \leq q_{max}$$

$$a' = \text{Inint}(q)$$

où f est une fonction monotone croissante, q_{max} le niveau de quantification maximum et Inint(q) l'entier le plus proche du niveau de quantification q.

2.4.1 Quantification linéaire

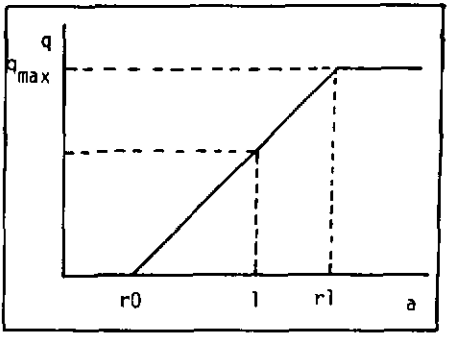


Fig.2.8 Quantification linéaire

Avant de définir la fonction q = f(a), remarquons d'abord que la valeur moyenne des éléments de la matrice à quantifier est égale à 1 (quelle que soit la normalisation d'amplitude).

Soit r0 la valeur, dans la matrice non quantifiée, à laquelle correspond la

valeur quantifiée nulle; et soit r1 la valeur, dans la matrice non quantifiée, à laquelle correspond la valeur quantifiée maximale q_{max}.

La valeur de q_{max} dépend uniquement du nombre maximum Nb de bits désiré : $q_{max} = 2^{Nb}-1$.

Nous définissons la fonction q = f(a) comme suit :

$$q = \begin{cases} 0 & a \leq r0 \\ \dfrac{2^{Nb}-1}{r1 - r0}\,(a - r0) & r0 < a \leq r1 \\ 2^{Nb}-1 & a > r1 \end{cases}$$

Cette dernière relation est indépendante du niveau d'amplitude des éléments de la matrice à la sortie de l'analyseur spectral. En effet les grandeurs a, r0 et r1 sont des valeurs rapportées à la valeur moyenne des éléments de la matrice d'entrée qui est égale à l'unité dans notre cas. Les paramètres à étudier sont donc r1 , r0 et Nb.

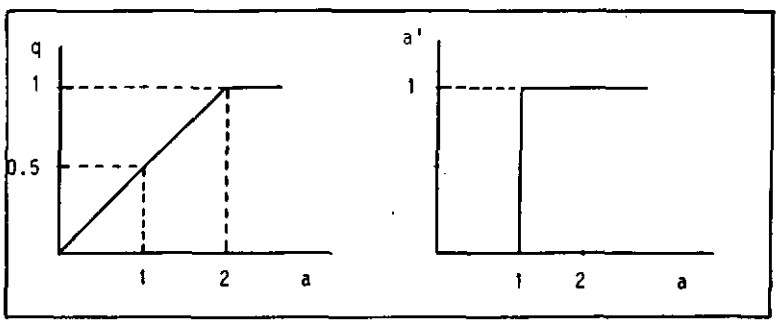


Fig.2.9 Exemple de quantification à 1 bit. (Nb = 1 ; r1 = 2 ; r0 = 0)

Lorsque Nb = 1 (quantification à 1 bit), toutes les droites de quantification qui se croisent en un point qui a pour ordonnée q = 0.5 sont équivalentes. Ainsi, parmi toutes les droites qui se croisent en un même point, il suffit de considérer celle dont le paramètre r0 est nulle. Par conséquent, lorsque Nb = 1, il suffit d'étudier le paramètre r1, r0 étant

fixé à zéro.

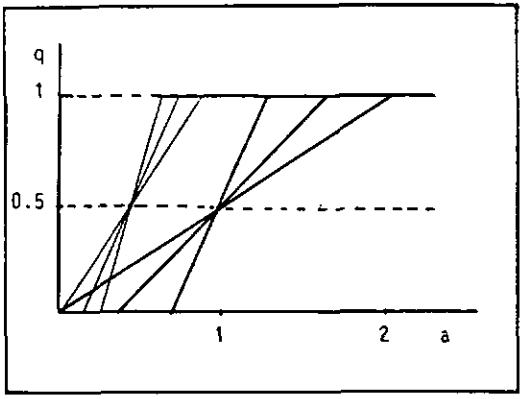


Fig.2.10

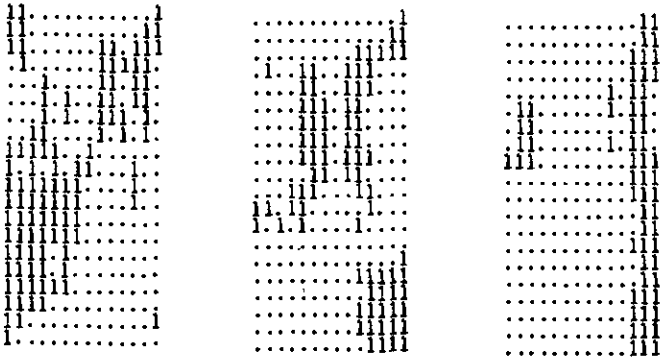


Fig.2.11 Exemples de matrices binaires obtenues après la compression temporelle linéaire, la normalisation en amplitude (Z3) et la quantification linéaire (Nb = 1, r0 = 0, r1 = 2).

2.4.2 La quantification logarithmique

Les grandeurs q_{max}, r0 et r1 ont la même définition que dans le paragraphe précédent. Par contre, la fonction f (q=f(a)) est logarithmique.

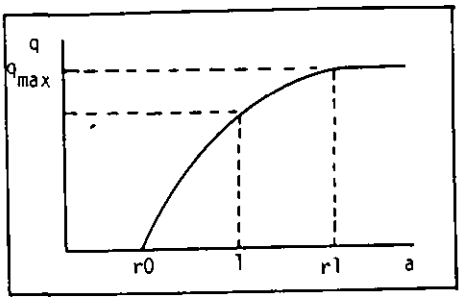


Fig.2.12 Quantification logarithmique

La fonction logarithmique est déterminée en choisissant deux points, par exemple les deux points (r0,0) et (r1,q_{max}).

$$q = \begin{cases} 0 & a \leq r0 \\ \dfrac{2^{Nb} - 1}{\ln(r1/r0)} \cdot \ln\left(\dfrac{a}{r0}\right) & r0 < a \leq r1 \\ 2^{Nb}-1 & a > r1 \end{cases}$$

Le rapport r1/r0 définit la dynamique que l'on peut couvrir.

2.5 Evaluation expérimentale des paramètres du traitement numérique

Nos expériences ont pour but d'évaluer l'effet des différents paramètres du traitement numérique sur les performances de reconnaissance en monolocuteur et multilocuteur. Nous avons une reconnaissance monolocuteur si le locuteur-référence est le même que le locuteur-test, et une reconnaissance multilocuteur si le locuteur-référence est différent du locuteur-test.

La distance entre deux mots est déterminée par l'algorithme de programmation dynamique (DTW) décrit dans le prochain chapitre.
Les paramètres à tester sont indiqués dans le tableau 2.1

Afin de rendre l'expérience aussi proche que possible de la réalité, aucune correction sur les mots enregistrés n'est effectuée. En effet, il arrive souvent qu'en raison d'une mauvaise prononciation par le locuteur, le début ou la fin d'un mot soit tronqué (par exemple 'quatre' qui devient 'atre' ou 'quat'). De même, aucune mesure n'a été prise contre le bruit provenant de l'environnement, sauf au niveau de la détection (§2.1).

2.5.1 Tests en reconnaissance monolocuteur

Soit l'expérience :

TST A : Le vocabulaire de test est composé de 13 mots comprenant les chiffres et trois mots de commande :

zéro	en-avant
un	en-arrière
deux	terminer
trois	
quatre	
cinq	
six	
sept	
huit	
neuf	

Le test de reconnaissance est effectué sur 5 locuteurs et 5 locutrices. Pour chacun des locuteurs, l'ensemble des mots inconnus est constitué de 3 prononciations de chaque mot du vocabulaire.
D'autre part, pour chacun des locuteurs, l'ensemble des mots de référence est constitué de la moyenne de 3 autres prononciations de chaque mot du

vocabulaire.
Nous obtenons ainsi un total de 390 essais de reconnaissance.

Paramètres du traitement numérique	
NTMP : Normalisation temporelle	
NTMP = 1	Normalisation linéaire
2	Normalisation curviligne
N	Longueur de normalisation
NAMP : Normalisation en amplitude	
NAMP = Z0	Norm. en amplitude globale
Z1	Norm. en amplitude par ligne
Z2	Norm. en amplitude par demi-colonne
Z3	Norm en amplitude par colonne
Z4	Norm. en amplitude par zone de longueur fréquentielle LF et de longueur temporelle LT
QUANT : Quantification	
QUANT = 1	Quantification linéaire
= 2	Quantification logarithmique
Nb	Nombre de bits
r0	Valeur, dans la matrice non quantifiée, à laquelle correspond la valeur quantifiée nulle. r0 est rapporté à la valeur moyenne de la matrice non quantifiée.
r1	Valeur, dans la matrice non quantifiée, à laquelle correspond la valeur quantifiée maximale. r0 est rapporté à la valeur moyenne de la matrice non quantifiée.

Tableau 2.1

Normalisation temporelle

Les conditions des tests de reconnaissance sont celles décrites dans TST A.

Les figures 2.13a, 2.13b et 2.13c donnent une évaluation expérimentale des normalisations temporelles linéaire (NTMP = 1) et curviligne (NTMP = 2), en fonction de différentes longueurs de normalisation.
Nous donnons les résultats obtenus pour trois types de normalisation en amplitude : a) Z0, b) Z2 et c) Z3.

Dans les trois figures 2.13a, 2.13b et 2.13c, nous observons clairement la supériorité de la normalisation curviligne par rapport à la normalisation linéaire quelle que soit la normalisation en amplitude (rappelons que la distance entre les mots est déterminée d'après l'algorithme de programmation dynamique DTW). Ce résultat est caractéristique de la reconnaissance monolocuteur. En effet, nous verrons plus loin que, dans la reconnaissance multilocuteur, la normalisation curviligne n'est pas supérieure à la normalisation linéaire.

Nous observons également que le taux d'erreur diminue très peu lorsque la longueur de compression temporelle est supérieure à vingt (il s'agit de vingt vecteurs-fréquences). Si l'objectif est de comprimer au maximum la quantité de données par mot, une longueur de compression égale à vingt (N = 20) est donc suffisante.

En conclusion, la reconnaissance monolocuteur des mots utilise avantageusement la compression temporelle curviligne. En outre, si l'on admet que la durée moyenne d'un mot est de 500 ms (cela donne 50 vecteurs-fréquences), nous pouvons comprimer la quantité de données d'un facteur 2.5 sans détérioration des performances.

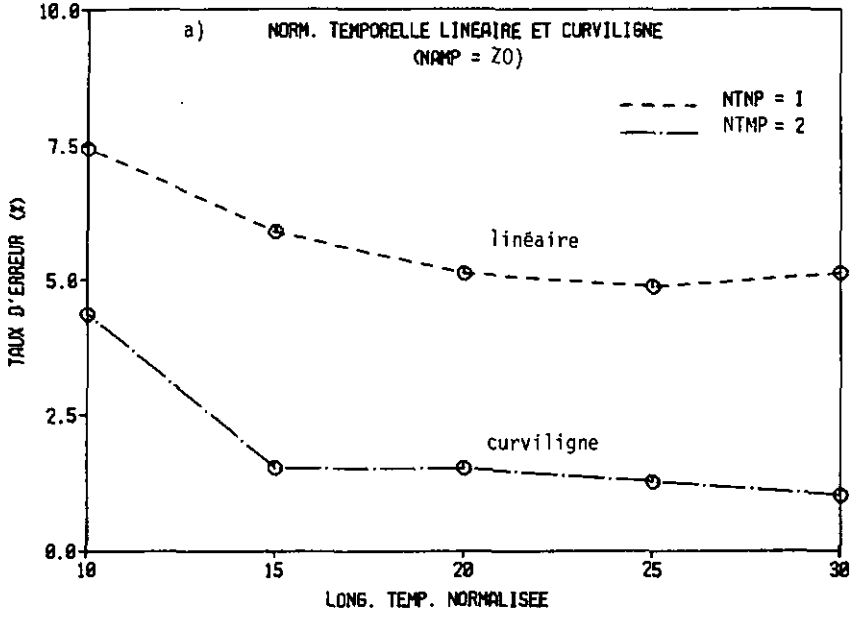


Fig.2.13a

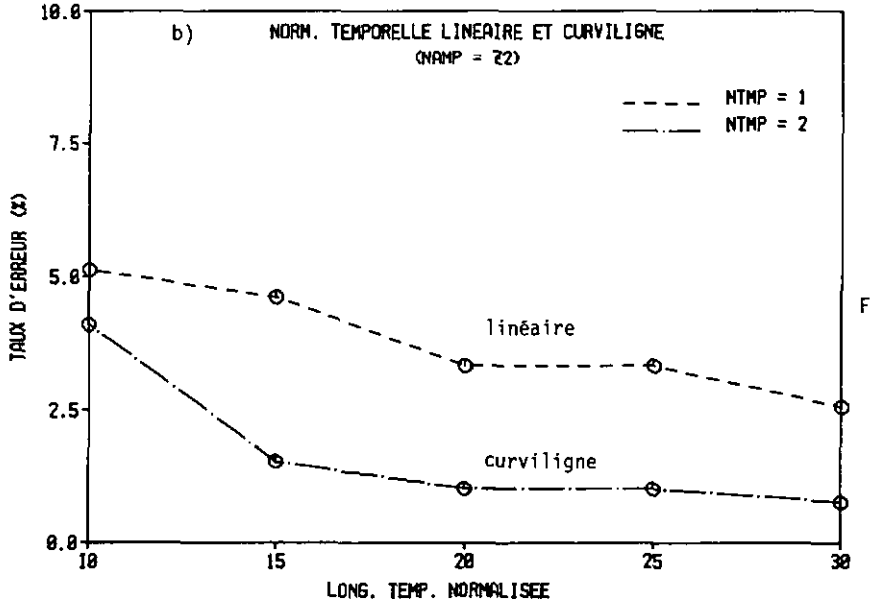


Fig.2.13b

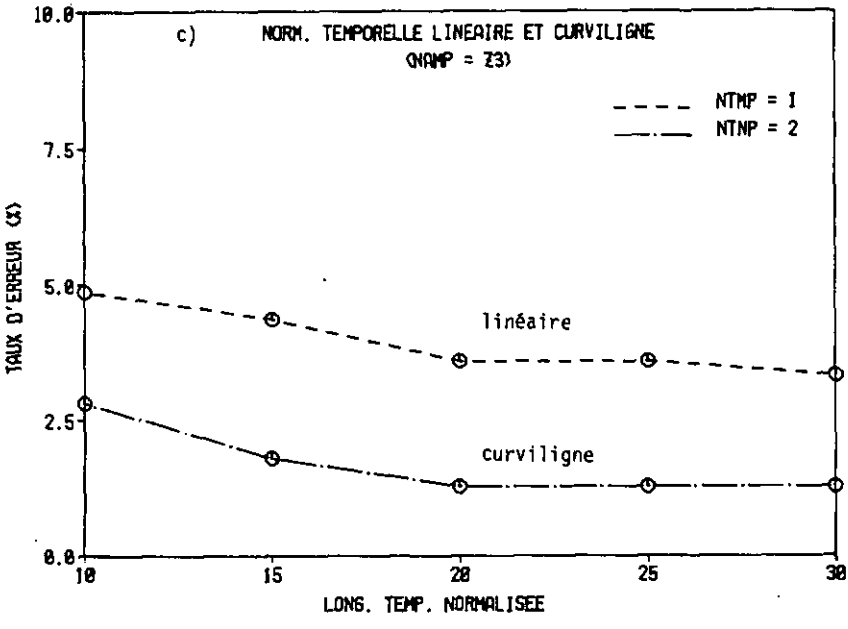


Fig.2.13c

Fig.2.13 Normalisation temporelle linéaire et curviligne : taux d'erreur en reconnaissance monolocuteur en fonction de la longueur de normalisation (TST A).

a) avec Z = Z0 ; b) avec Z = Z2 ; c) avec Z = Z3

Choix des paramètres (Annexe A) :

NTMP = Ø N = Ø
NAMP = Z0,Z2,Z3 LF = - LT = -
QUANT = - Nb = - r0 = - r1 = -
DMAT = 3 OVEC = 2 RANGE = N/5 CDF = 1

N0 :

Ø : paramètre à l'étude
- : paramètre non considéré

Normalisation en amplitude

Les conditions des tests sont celles décrites dans TST A

Les résultats de l'évaluation expérimentale des différentes méthodes de normalisation en amplitude sont donnés dans les figures 2.14a et 2.14b.

La figure 2.14a compare les performances de reconnaissance des 4 cas de normalisation en amplitude suivants :

1) normalisation en amplitude globale (Z0)
2) normalisation en amplitude d'après la ligne (Z1)
3) normalisation en amplitude d'après la demi-colonne (Z2)
4) normalisation en amplitude d'après la colonne (Z3)

Parmi les quatre méthodes testées (Fig.2.14a), la normalisation en amplitude d'après la ligne (Z1) donne les plus mauvais résultats. La normalisation en amplitude globale (Z0) est moins favorable que la normalisation en amplitude d'après la demi-colonne (Z2) ou d'après la colonne (Z3).

La normalisation en amplitude d'après la demi-colonne (Z2) qui consiste à effectuer un traitement séparé des canaux basses fréquences et hautes fréquences pour éviter un étouffement des motifs hautes fréquences donne des résultats similaires à Z3. La normalisation d'après Z2 est intéressante surtout si la préaccentuation des hautes fréquences s'avère insuffisante (en effet le niveau d'énergie des composantes hautes fréquences du signal de parole à la sortie du microphone est très faible).

La figure 2.14b donne le taux d'erreur en % obtenu pour la normalisation en amplitude d'après la zone Z4. Les dimensions LF et LT de Z4 (qui est une sous-matrice de A) sont les paramètres à tester.

Nous observons que les meilleurs résultats sont obtenus quand la longueur fréquentielle LF est grande : LF doît être supérieure à 7 (LF > K/2). Dans ce cas là (LF> K/2), les meilleurs résultats sont obtenus quand la longueur temporelle LT est petite (LT = 1, 3, 5).
En résumé, dans une normalisation en amplitude d'après la zone rectangulaire (Z4), les meilleurs résultats sont obtenus pour une longueur fréquentielle grande et pour une longueur temporelle relativement petite.
Par ailleurs, les résultats obtenus par la normalisation en amplitude d'après Z4(LF,LT), lorsque les dimensions LF et LT sont judicieusement choisies, sont meilleurs par rapport à une simple normalisation en amplitude globale.

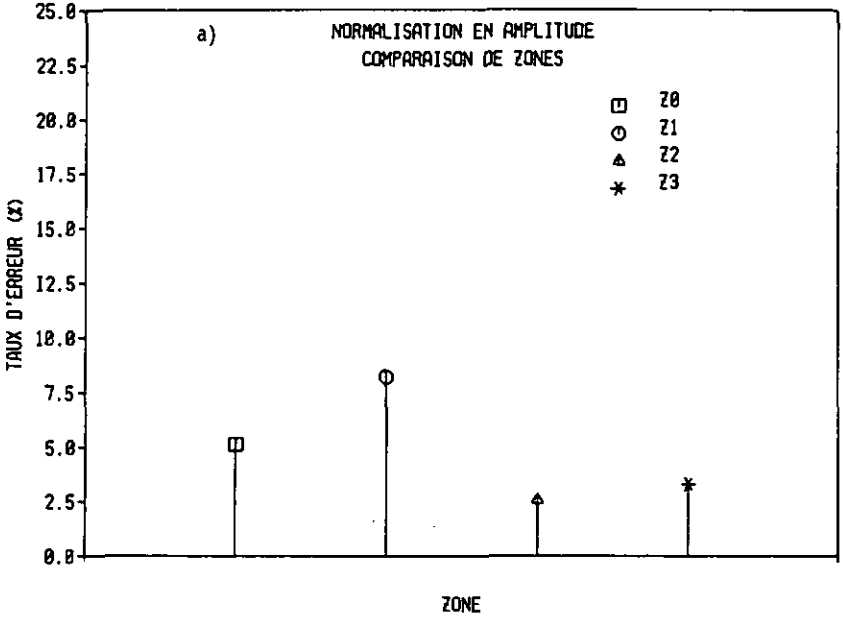

a)
NORMALISATION EN AMPLITUDE
COMPARAISON DE ZONES
Z0
Z1
Z2
Z3
TAUX D'ERREUR (%)
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
ZONE

Fig.2.14a

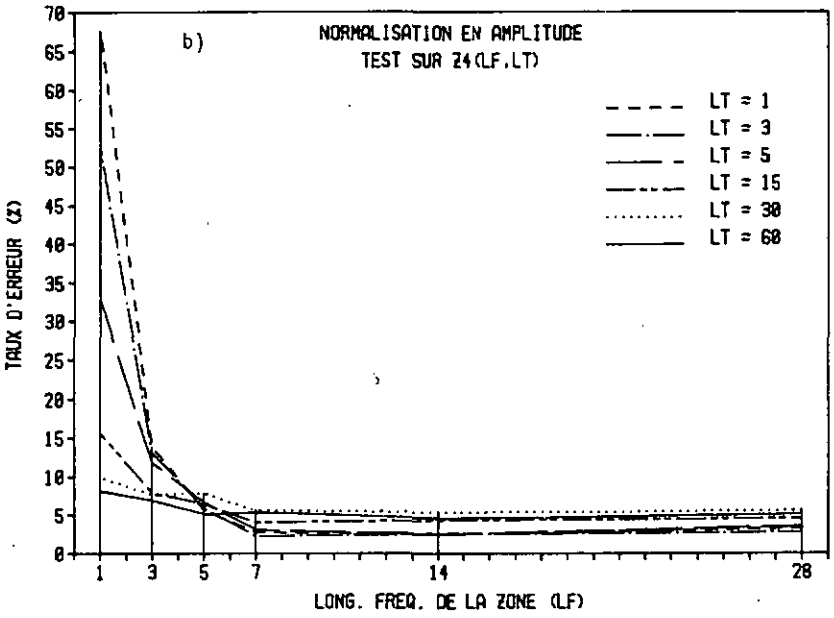


Fig.2.14b

Fig.2.14 Normalisation en amplitude : taux d'erreur en reconnaissance monolocuteur en fonction du choix de la zone (TST A).

a) Z0, Z1, Z2 et Z3
b) Z4(LT,LF)

Choix des paramètres :

NTMP = 1 N = 30
NAMP = ✓ LF = ✓ LT = ✓
QUANT = - Nb = - r0 = - r1 = -
DMAT = 3 DVEC = 2 RANGE = 6 COF = 1

Quantification linéaire

Les conditions des tests de reconnaissance sont celles décrites dans TST A.

Rappelons que le but que nous nous fixons en effectuant la quantification est de réduire la quantité de données sans détériorer les performances.

Les figures 2.15a, 2.15b, 2.15c et 2.15d donnent les résultats de l'évaluation expérimentale des différents paramètres de la quantification linéaire : nombre de bits Nb, facteur r1 et facteur r0.

Afin de bien interpréter les résultats, signalons qu'une valeur du facteur r0 égale à un (r0=1) signifie que tous les éléments de la matrice d'entrée (obtenue après la normalisation en amplitude) qui sont inférieurs à la valeur moyenne des éléments de la matrice d'entrée sont mis à zéro dans la matrice de sortie.
De façon similaire, une valeur du facteur r1 égale à un (r1=1) signifie que tous les éléments de la matrice d'entrée qui sont supérieurs à la valeur moyenne des éléments de la matrice sont mis à la valeur maximale quantifiée (2^{Nb}-1) dans la matrice de sortie.

Ainsi, les vecteurs-fréquences des matrices obtenues après quantification dépendent non seulement de la courbe de quantification, mais aussi du type de normalisation en amplitude.

Les figures 2.15a et 2.15b donnent le taux d'erreur en fonction du nombre de bits Nb pour différentes valeurs du facteur r1. Le facteur r0 est fixé à zéro. Dans la Fig.2.15a c'est la compression temporelle linéaire qui est appliquée; dans la figure 2.15b c'est la compression temporelle curviligne linéaire qui est appliquée.

Nous avons les résultats suivants :

- lorsque le nombre de bits est supérieur à trois, le taux d'erreur ne diminue plus.

- lorsque le nombre de bits est petit ($Nb \leq 3$), le taux d'erreur dépend fortement du choix de r1.

En conclusion, si on choisit judicieusement le facteur r1, le taux d'erreur obtenu pour un petit nombre de bits est comparable au taux d'erreur obtenu pour un grand nombre de bits. Ce résultat est important dans la perspective d'une compression maximale des références pour la miniaturisation du système de reconnaissance.

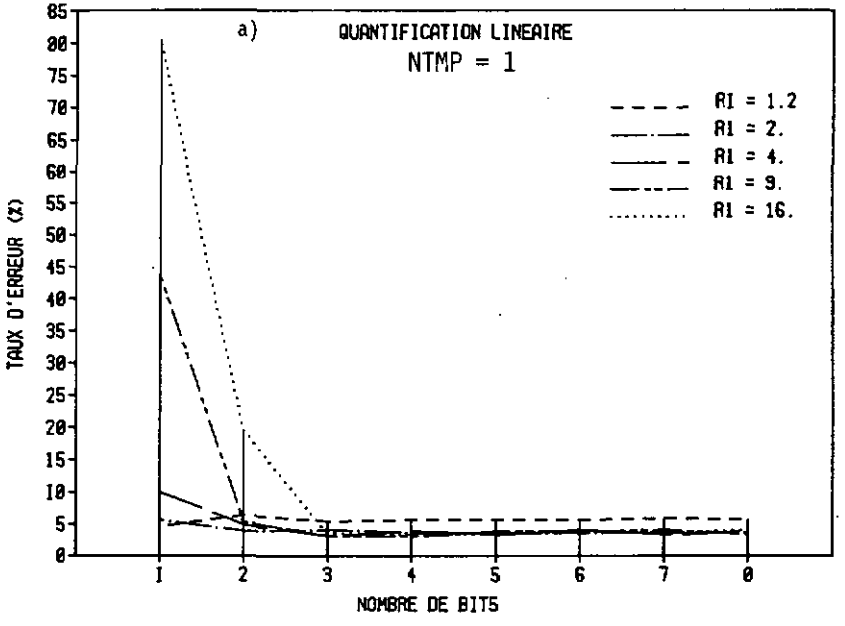

a)
QUANTIFICATION LINEAIRE
NTMP = 1
RI = 1.2
RI = 2.
RI = 4.
RI = 9.
RI = 16.
TAUX D'ERREUR (%)
NOMBRE DE BITS
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
1
2
3
4
5
6
7
0

Fig.2.15a

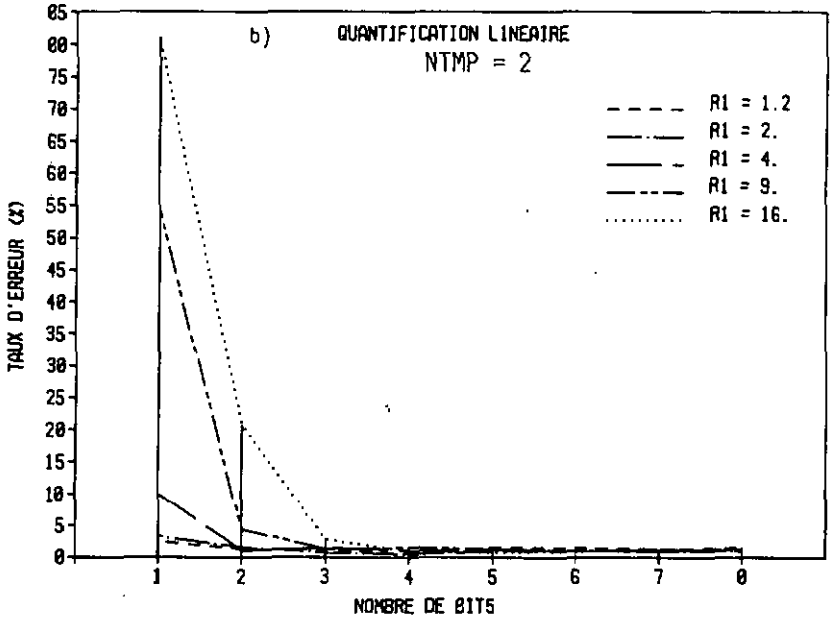


Fig.2.15b

Fig.2.15 Quantification linéaire : taux d'erreur en reconnaissance monolocuteur en fonction du nombre de bits Nb et du facteur r1. Le facteur r0 est fixé à zéro (TST A).

a) NTMP = 1 ; b) NTMP = 2

Choix des paramètres :

NTMP = 1,2 N = 20
NAMP = Z3 LF = - LT = -
QUANT = 1 Nb = Ø r0 = 0 r1 = Ø
DMAT = 3 DVEC = 2 RANGE = 4 COF = 1

La figure 2.15c montre l'effet du choix du facteur r0 sur le taux d'erreur, dans les cas d'une quantification avec un nombre de bits égal à huit.

Nous observons que le choix de r0, pour autant qu'il soit inférieur à 0.5, a peu d'influence sur le taux d'erreur.

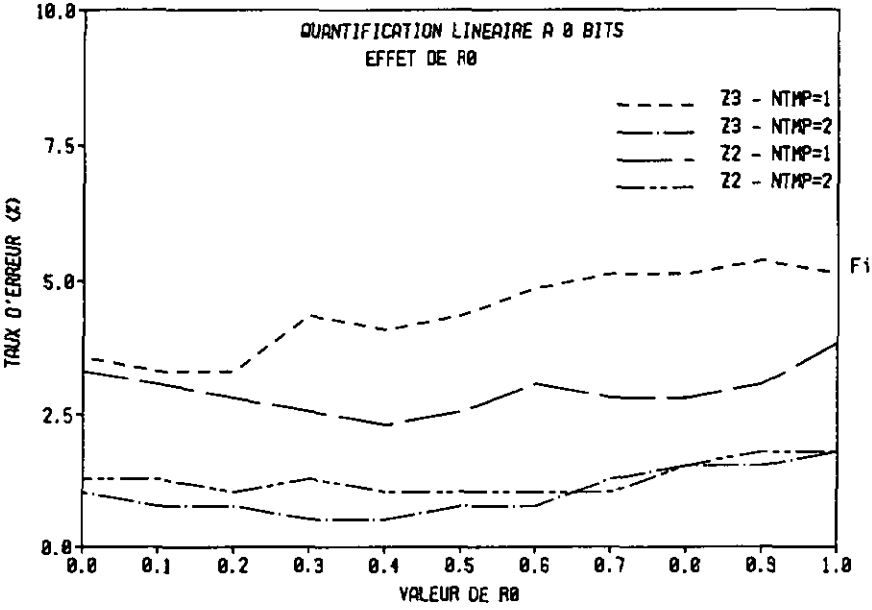


Fig.2.15c

Fig.2.15c Quantification linéaire : effet du facteur r0 sur le taux d'erreur en reconnaissance monolocuteur (TST A).

Choix des paramètres :

NTMP = 1,2 N = 20
NAMP = Z2,Z3 LF = - LT = -
QUANT = 1 Nb = 8 r0 = ~~0~~ r1 = 2
DMAT = 3 DVEC = 2 RANGE = 4 CDF = 1

Nous avons vu dans les figures 2.15a et 2.15b qu'il est possible, en choisissant judicieusement le facteur r1, de diminuer le nombre de bits sans trop détériorer les performances.

Lorsque le nombre de bits est égal à un (Nb = 1), il suffit d'étudier l'influence du choix de r1 sur le taux d'erreur car à toute variation de r0 nous pouvons associer une variation équivalente de r1 (§2.4.1). Nous fixerons donc r0 = 0 lorsque Nb = 1.

La figure 2.15d évalue expérimentalement le domaine dans lequel nous pouvons choisir le facteur r1 dans le cas d'une quantification à un bit. Différents cas de normalisation en amplitude et normalisation temporelle sont testés.

Nous observons qu'un choix de r1 tel que :

$$\underline{1 < r1 < 3}$$

donne des résultats tout à fait comparables au cas où il n'y a aucune quantification. Ainsi, grâce à un choix judicieux des paramètres de quantification, nous pouvons réduire très fortement la quantité de données.

Remarque : une normalisation en amplitude d'après Z3 et une quantification avec les paramètres suivants : Nb = 1, r0 = 0 et r1 = 2, reviennent à poser égal à 1 tout élément a(k,m) de la matrice qui est supérieur à la valeur moyenne des composantes de $\underline{a}_m$, et égal à 0 tout élément qui est inférieur ou égal à la valeur moyenne des composantes de $\underline{a}_m$.

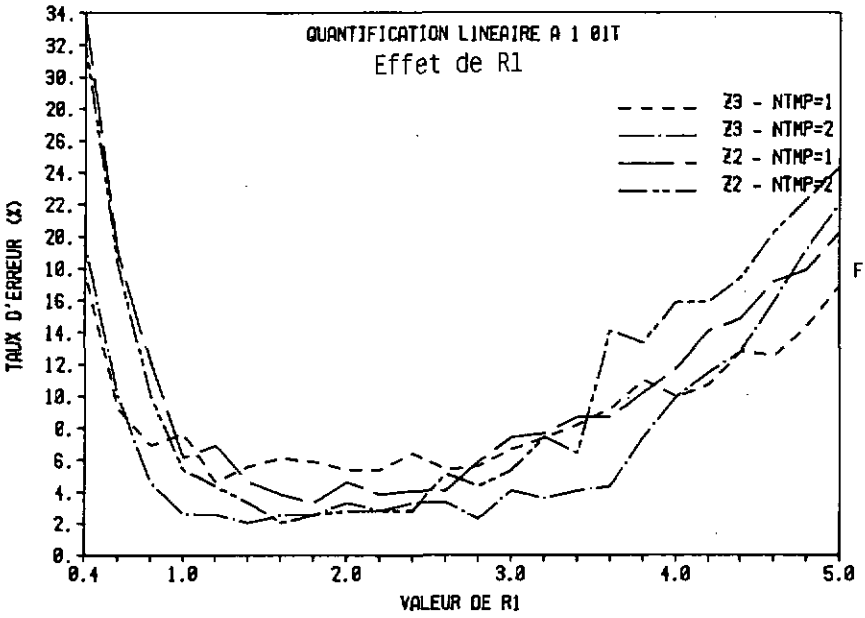


Fig.2.15d

Fig.2.15d Quantification linéaire à 1 bit : effet du facteur r1 sur le taux d'erreur en reconnaissance monolocuteur (TST A).

Choix des paramètres :

NTMP = 1,2 N = 20

NAMP = Z2,Z3 LF = - LT = -

QUANT = 1 Nb = 1 r0 = 0 r1 = Ø

DMAT = 3 DVEC = 2 RANGE = 4 COF = 1

Quantification logarithmique

Les conditions des tests sont celles décrites dans TST A

La figure 2.16 donnent une évaluation expérimentale des différents paramètres de la quantification logarithmique : nombre de bits Nb, facteur r1 et facteur r0 (ou dynamique r1/r0). Nous observons que globalement, la quantification logarithmique n'apporte rien de plus par rapport à la quantification linéaire. Il convient donc d'utiliser la quantification linéaire, d'autant plus que le calcul est plus facile et plus rapide.

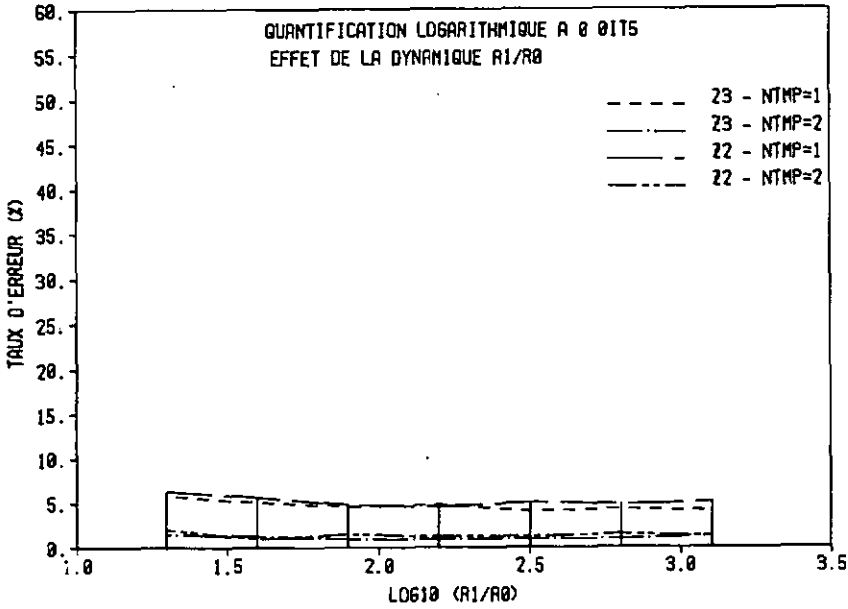


Fig.2.16 Quantification logarithmique : taux d'erreur en reconnaissance monolocuteur en fonction du facteur rI (TST A).

Choix des paramètres :

NTMP = 1,2 N = 20

NAMP = 23 LF = - LT = -

QUANT = 2 Nb = 8 r0 =0.1 r1 = 0

DMAT = 3 DVEC = 2 RANGE = 4 CDF = 1

2.5.2 Tests en reconnaissance multilocuteur

Les tests présentés dans ce paragraphe permettent de savoir si un environnement multilocuteur implique un choix différent des paramètres du traitement numérique.

TST B : Le vocabulaire de test est le même que dans TST A.
Le test est effectué sur 5 locuteurs et 5 locutrices. Le locuteur-test est toujours différent du locuteur-référence.
Pour chacun des locuteurs, l'ensemble des mots de référence est constitué de la moyenne de 3 prononciations de chaque mot du vocabulaire.
Nous effectuons un total de 585 essais de reconnaissance multilocuteur.

Les résultats des tests présentés dans les figures 2.17 à 2.19 résument l'effet des différents paramètres du traitement numérique sur le taux d'erreur dans un environnement multilocuteur.

Normalisation temporelle

Les conditions des tests sont celles décrites dans TST B

Les figures 2.17a, 2.17b et 2.17c donnent l'évaluation expérimentale des normalisations temporelles linéaire et curviligne, en fonction de différentes longueurs de normalisation, dans un environnement multilocuteur.

Nous observons que, contrairement au cas monolocuteur, la normalisation temporelle linéaire est supérieure à la normalisation temporelle curviligne. D'autre part, une longueur de compression temporelle normalisée égale à vingt (N = 20) est suffisante.

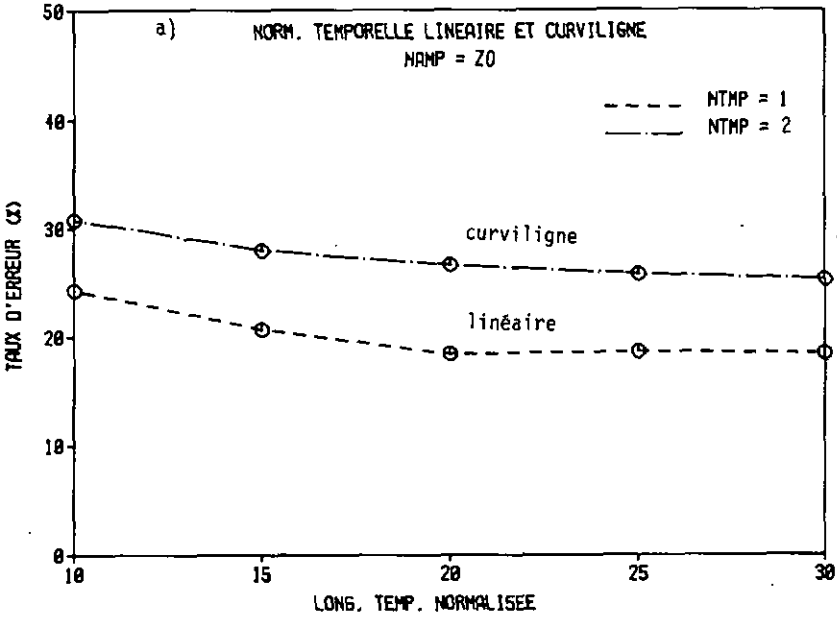


Fig.2.17a

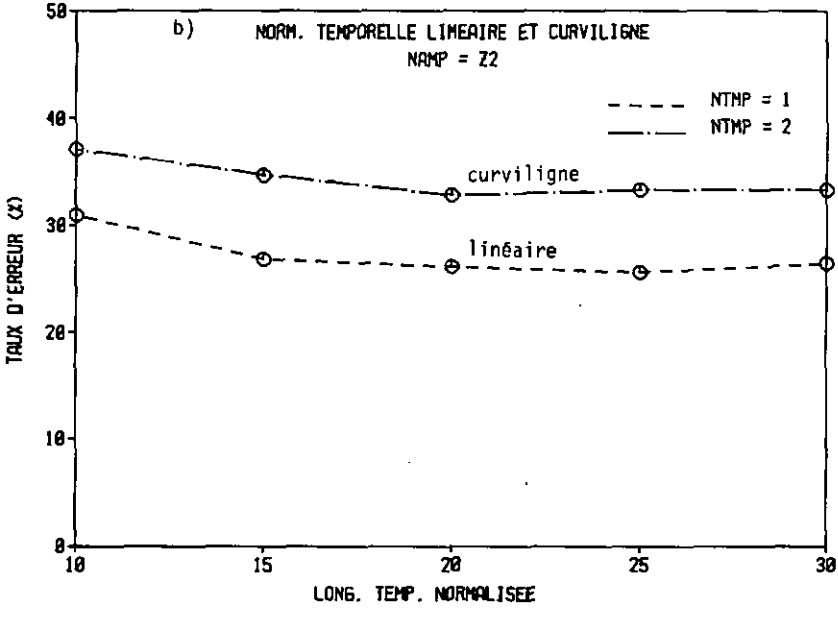


Fig.2.17b

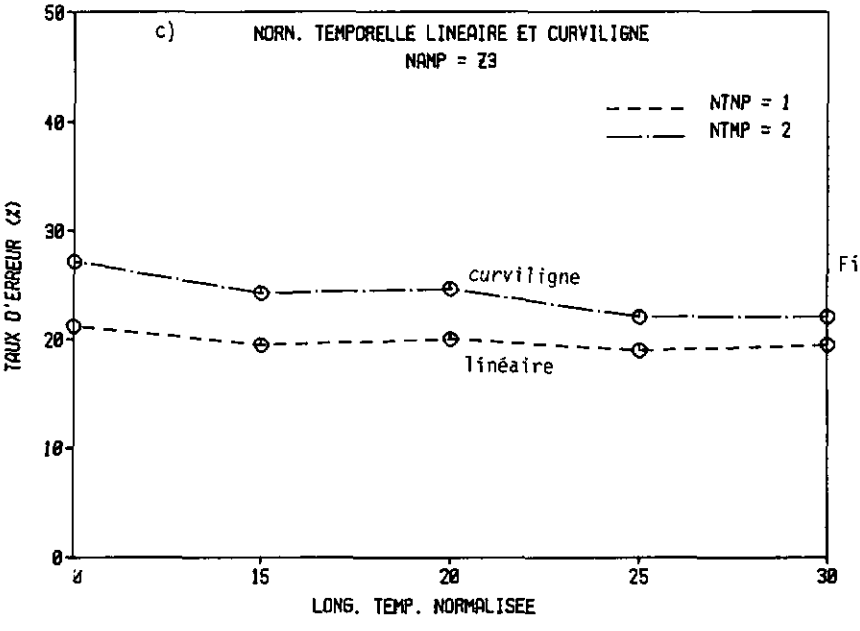


Fig.2.17c

Fig.2.17 Normalisation temporelle linéaire et curviligne : taux d'erreur en reconnaissance multilocuteur en fonction de la longueur de normalisation (TST B).

a) avec Z = Z0 ; avec Z = Z2 ; c) avec Z = Z3

Choix des paramètres :

NTMP = Ø N = Ø

NAMP = Z0, Z2, Z3 LF = - LT = -

QUANT = - Nb = - r0 = - r1 = -

DMAT = 3 DVEC = 2 RANGE = 4 CDF = 1

Normalisation en amplitude

Les conditions des tests sont celles décrites dans TST B

La figure 2.18a donne les résultats de l'évaluation de la normalisation en amplitude d'après Z0, Z1 Z2, et Z3; la figure 2.18b donne les résultats de l'évaluation d'après Z4(LF,LT)

Les performances obtenues par la normalisation en amplitude globale (Z0) sont supérieures à celles obtenues par Z1 et Z2, et ils sont pratiquemment les mêmes que ceux obtenus par Z3.

Dans le cas de la normalisation en amplitude d'après Z4 (fig.2.18b), nous observons que les différents choix de la longueur fréquentielle (LF) tendent aux mêmes comportements qu'en reconnaissance monolocuteur. Par contre, contrairement au cas monolocuteur, une zone plus large de la longueur temporelle (LT) donne des résultats plus favorables. C'est pourquoi les résultats obtenus par la normalisation en amplitude globale sont comparables aux meilleurs résultats obtenus par la normalisation d'après Z4.

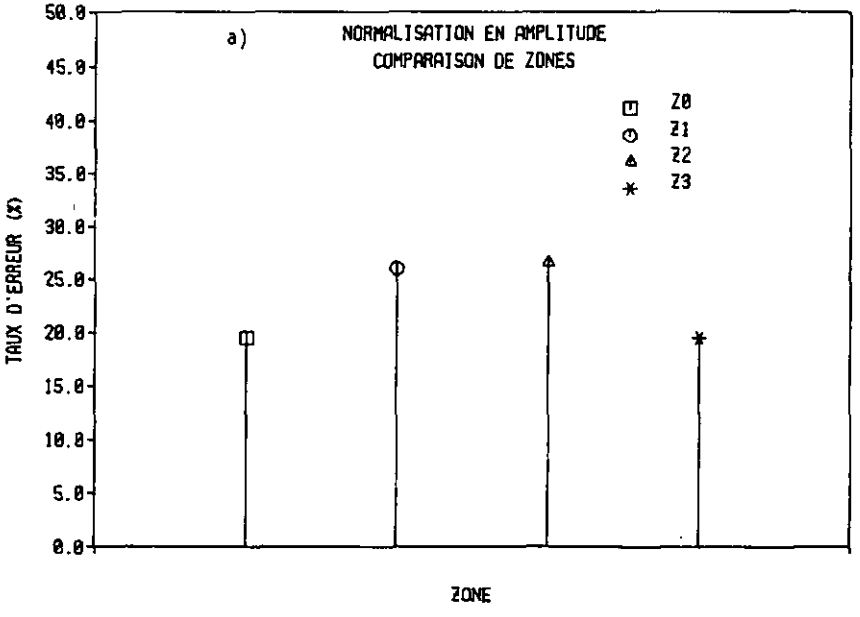


Fig.2.18a

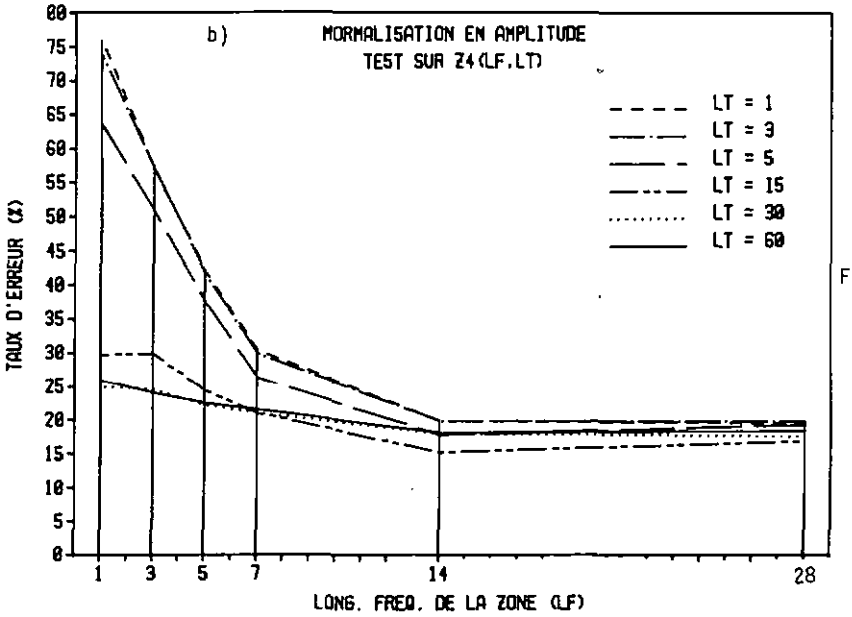


Fig.2.18b

Fig.2.18 Normalisation en amplitude : taux d'erreur en reconnaissance multilocuteur en fonction du choix de la zone (TST 8).

a) Z0, Z1, Z2 et Z3 ; b) Z4(LT,LF)

Choix des paramètres

NTMP = 1 N = 30

NAMP = Ø LF = Ø LT = Ø

QUANT = - Nb = - r0 = - r1 = -

UMAT = 3 DVEC = 2 RANGE = 6 COF = 1

Quantification

Les figures 2.19a et 2.19b donnent le taux d'erreur en fonction du nombre de bits Nb pour différentes valeurs du facteur r1. Le facteur r0 est fixé à zéro. Dans la Fig.2.19a c'est la compression temporelle linéaire (NTMP = 1) qui est appliquée, alors que dans la Fig.2.19b c'est la compression temporelle curviligne (NTMP = 2) qui est appliquée.

Comme dans le cas monolocuteur, nous observons que le taux d'erreur est indépendant du nombre de bits lorsque celui-ci est supérieur à trois. Si le nombre de bits est petit ($Nb \leq 3$), le taux d'erreur dépend fortement du choix de r1.

Ainsi, comme dans le cas monolocuteur, si l'on choisit judicieusement le facteur r1, le taux d'erreur obtenu pour un nombre faible de bits est comparable au taux d'erreur obtenu pour un plus grand nombre de bits.

En résumé, le comportement des paramètres de la quantification linéaire tendent aux mêmes comportements en reconnaissance monolocuteur et multilocuteur.

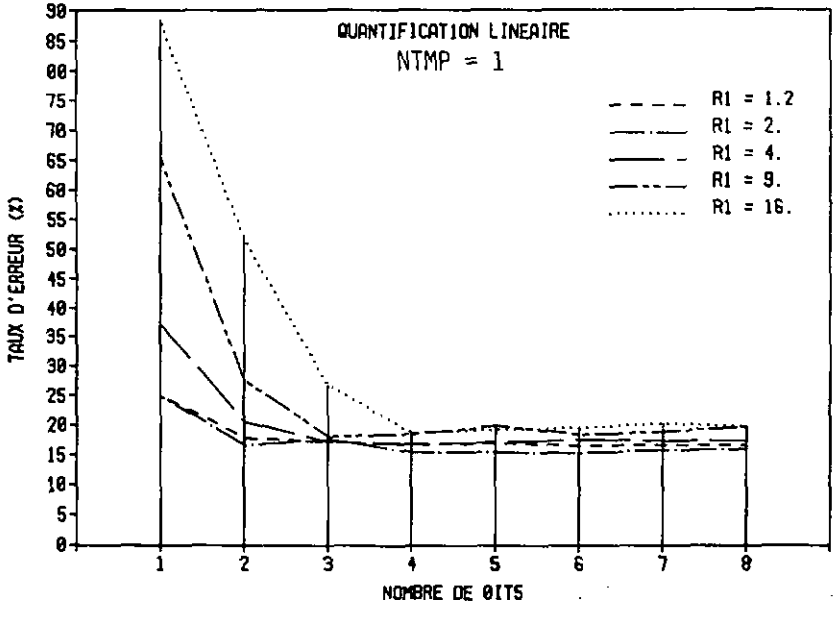


Fig.2.19a

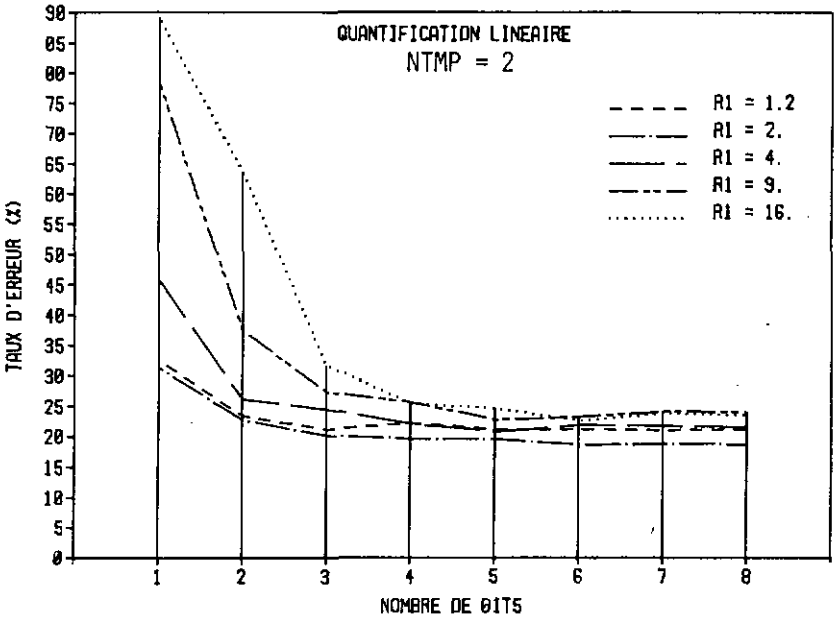


Fig.2.19b

Fig.2.19 Quantification linéaire : taux d'erreur en reconnaissance multilocuteur en fonction du nombre de bits Nb et du facteur r1. Le facteur r0 est fixé à zéro (TST 0).

a) NTMP = 1 ; b) NTMP = 2

Choix des paramètres

NTMP = 1,2 N = 20
NAMP = Z3 LF = - LT = -
QUANT = 1 Nb = Ø r0 = 0 r1 = Ø
DMAT = 3 DVEC = 2 RANGE = 4 CDF = 1

2.6 Discussion

Pour un système de reconnaissance miniaturisé, l'un des facteurs critiques est la capacité en mémoire.

La capacité en mémoire exigé est proportionnelle au nombre de références et au nombre total de bits par mot de référence K.N.Nb.

Le nombre K de canaux n'a pas fait l'objet d'une étude particulière dans ce travail.

Le facteur N a pu être ramené d'une valeur moyenne de 50 vecteurs-fréquences (durée moyenne d'un mot égale à 500 ms) à 20 vecteurs-fréquences grâce à la compression temporelle linéaire ou curviligne; et ceci sans dégradation des performances.

Quant au facteur Nb, grâce à une combinaison judicieuse de la normalisation en amplitude et de la quantification, il a pu être ramené d'une valeur de 12 bits à 1 bit; et ceci sans détérioration des performances.

Ainsi, par un traitement approprié, nous pouvons diminuer de façon considérable la quantité de données obtenue à la sortie de l'analyseur spectral. La taille en mémoire exigée diminuant de façon similaire, la miniaturisation devient plus aisée.

3. Calcul de la distance entre mots

Le but du classifieur est d'identifier un mot inconnu à l'un des mots du dictionnaire. La décision est prise à la suite du calcul de la dissimilitude - ou distance - entre le mot inconnu et chacun de mots du dictionnaire.

La prononciation d'un mot donné est, en général, composée de plusieurs événements vocaux (phonèmes, transitions entre phonèmes, silence avant les plosives, etc). La durée d'un même événement vocal varie d'une prononciation à l'autre.

Lors du calcul de la distance entre deux matrices, l'une représentant le mot inconnu et l'autre un mot de référence, il faudrait obtenir une petite distance (idéalement une distance nulle) lorsqu'il s'agit du même mot. Ceci n'est possible que si l'on arrive à faire coïncider les mêmes événements vocaux, représentés dans les deux matrices.

Nous présentons dans ce chapitre les différentes méthodes utilisées pour effectuer le calcul de distance entre deux mots. Les méthodes diffèrent essentiellement par leur capacité à faire coïncider les mêmes événements vocaux des différentes prononciations d'un mot donné.

Soit $\underline{A}(K,M)$ et $\underline{B}(K,N)$ deux matrices représentant deux mots à comparer. $\underline{A}$ et $\underline{B}$ sont représentées comme suit :

$$\underline{A}(K,M) = (\underline{a}_1, \underline{a}_2, \ldots, \underline{a}_m, \ldots, \underline{a}_M)$$

$$\underline{B}(K,N) = (\underline{b}_1, \underline{b}_2, \ldots, \underline{b}_n, \ldots, \underline{b}_N)$$

où $\underline{a}_m$ et $\underline{b}_n$ sont des vecteurs-fréquences :

$$\underline{a}_m = (a_m(1), a_m(2), \ldots, a_m(k), \ldots, a_m(K))$$

$$\underline{b}_n = (b_n(1), b_n(2), \ldots, b_n(k), \ldots, b_n(K))$$

3.1 Distance entre deux vecteurs-fréquences

Soit $d(m,n) = d(\underline{a}_m, \underline{b}_n)$ la distance entre deux vecteurs-fréquences $\underline{a}_m$ et $\underline{b}_n$. Les deux définitions suivantes sont utilisées par la suite pour déterminer la distance entre deux matrices :

1) distance absolue (ou "City block")

$$d1(m,n) = \sum_{k=1}^{K} | a_m(k) - b_n(k) | \qquad (3.1)$$

2) distance relative (ou distance de Gamberra) /15/

$$d2(m,n) = \frac{\sum_{k=1}^{K} | a_m(k) - b_n(k) |}{\sum_{k=1}^{K} (a_m(k) + b_n(k))} \qquad (3.2)$$

3.2 Distance linéaire entre matrices

3.2.1 Distance simple

C'est la méthode la plus simple pour déterminer la dissimilitude entre deux prononciations. Elle suppose qu'il n'y a pas de fluctuations temporelles entre deux prononciations d'un mot donné. Elle est définie pour deux matrices $\underline{A}(K,N)$ et $\underline{B}(K,N)$ de même longueur N (c'est-à-dire contenant le même nombre de vecteurs-fréquences). La compression à la même longueur N des deux matrices est obtenue préalablement par la normalisation temporelle linéaire ou curviligne (§2.2). On a :

$$D(\underline{A},\underline{B}) = \frac{1}{N} \sum_{n=1}^{N} d(n,n) \qquad (3.3)$$

3.2.2 Distance linéaire avec déplacement global

Lorque le début ou la fin d'un mot est perdu (par exemple : perte d'une plosive en fin de mot), ou lorsque "s'ajoute" un bruit quelconque à son début ou à sa fin (par exemple : claquement de lèvres), le calcul de la distance selon la méthode précédente est nettement insuffisant. C'est pourqoui on définit la "distance linéaire avec déplacement global" pour deux matrices $\underline{A}(K,N)$ et $\underline{B}(K,N)$ de même longueur N.

La distance entre les matrices $\underline{A}$ et $\underline{B}$ est alors définie comme suit :

$$D(\underline{A},\underline{B}) = \min_{r} O(r) \qquad r = -R , \ldots , +R \qquad (3.4)$$

R : déplacement global maximum

Avec D(r) défini comme suit :

$$O(r) = \frac{1}{N} \sum_{n=N_{min}}^{N_{max}} d(n,n+r) \qquad (3.5)$$

où r est le déplacement global entre les deux matrices $\underline{A}$ et $\underline{B}$, et

$$\begin{aligned} N_{min} &= \text{Max } (1,1-r) \\ N_{max} &= \text{Min } (N,N-r) \end{aligned} \qquad (3.6)$$

On pose :

si : $n \leq 0$ ou $n > N$ alors : $\underline{a}_n = \underline{0}$ et $\underline{b}_n = \underline{0}$ (3.7)

Par exemple, un déplacement global de r= +2 consisterait à sommer les distances entre les vecteurs-fréquences suivants :

$\underline{0}$	$\underline{0}$	$\underline{a}_1$	$\underline{a}_2$	...	$\underline{a}_{N-1}$	$\underline{a}_N$
$\underline{b}_1$	$\underline{b}_2$	$\underline{b}_3$	$\underline{b}_4$	...	$\underline{0}$	$\underline{0}$

3.3 Distance dynamique entre deux matrices /7/, /36/, /57/, /78/, /84/

Les deux méthodes de calcul de distance des paragraphes 3.2.1 et 3.2.2 effectuent un alignement temporel linéaire, et sont donc insuffisantes dans

le cas de fortes distorsions entre les événements vocaux de deux prononciations différentes d'un mot donné.

C'est pourquoi, on utilise un alignement temporel non-linéaire : on établit une correspondance entre la séquence des $\underline{a}_m$ et des $\underline{b}_n$, de manière à faire correspondre à chaque fois un $\underline{a}_m$ et un $\underline{b}_n$ les plus semblables possible.

L'algorithme utilisé pour réaliser cet alignement temporel non-linéaire est basé sur le principe de la programmation dynamique exposé brièvement ci-après.

Le modèle mathématique de la programmation dynamique doit être complété par des contraintes liées aux structures linguistiques de la parole.

Alignement dynamique

Considérons le plan constitué par les deux axes temporels m et n. Les vecteurs $\underline{a}_m$ et $\underline{b}_n$ sont représentés par leurs indices le long des axes m et n (Fig.3.1). Soit p(i) un couple de deux vecteurs-fréquences $\underline{a}_{m(i)}$ et $\underline{b}_{n(i)}$, i étant la i-ème correspondance :

$$p(i) = (m(i),n(i)) \qquad (3.8)$$

L'alignement temporel non-linéaire - ou dynamique - est décrit par la séquence de couples suivante :

$$P = p(1),p(2),p(3),\ldots,p(i),\ldots,p(I) \qquad (3.9)$$

La séquence P est la fonction d'alignement (ou chemin d'alignement) qui réalise une correspondance entre les éléments $\underline{a}_m$ du mot A et $\underline{b}_n$ du mot B. I est le nombre de couples mis en correspondance.
S'il n'y a pas de distorsion temporelle entre les événements vocaux de deux prononciations d'un mot donné, cette fonction coïncide avec la diagonale m(i) = n(i) (Fig.3.1).

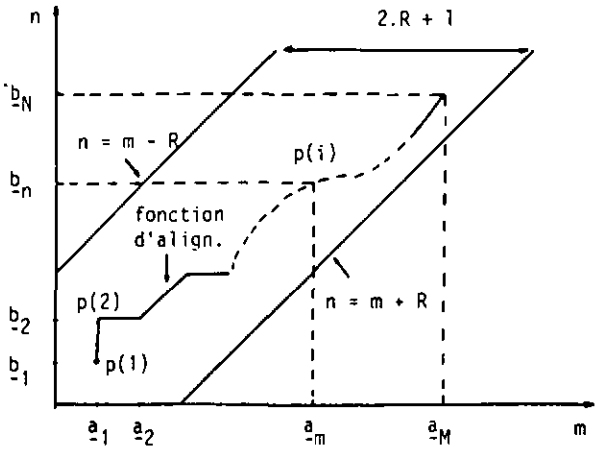


Fig.3.1 Alignement temporel dynamique

Soit

$$d(p(i)) = d(m(i),n(i)) = d(\underline{a}_{m(i)},\underline{b}_{n(i)}) \qquad (3.10)$$

la distance entre le couple de vecteurs-frequences $\underline{a}_m$ et $\underline{b}_n$ (voir §3.1), et soit E(P) la somme pondérée de ces distances le long du chemin P :

$$E(P) = \sum_{i=1}^{I} d(p(i)) \cdot w(i) \qquad (3.11)$$

où w(i) est une fonction de pondération que l'on précisera plus loin. E(P) atteint sa valeur minimale quand l'alignement temporel est réalisé de façon optimale. Ainsi, on définit la distance entre les mots A et B par la relation suivante :

$$D(A,B) = \min_{P} \frac{\sum_{i=1}^{I} d(p(i)).w(i)}{\sum_{i=1}^{I} w(i)} \qquad (3.12)$$

où $\sum_{i} w(i)$ est utilisé pour compenser l'effet du nombre de points le long du chemin d'alignement temporel P. Le problème posé par la relation (3.12) peut se ramener à un problème de programmation dynamique.

Application du théorème d'optimalité de la programmation dynamique

La programmation dynamique est une méthode d'optimisation applicable à de nombreux problèmes avec ou sans contraintes possèdant une propriété de décomposabilité /57/. Le point de départ de cette méthode est le théorème d'optimalité. Ce théorème peut être énoncé comme suit :

"Tout chemin optimal est constitué de sous-chemins optimaux".

Ce théorème d'optimalité s'applique aisément au problème d'alignement temporel. Supposons que le dénominateur de la relation (3.12) est indépendant du chemin P. La fonction D(A,B) des I couples p(1), p(2), ..., p(I) devient une fonction décomposable /57/. Ainsi :

$$\min_{p(1),\ldots,p(I)} \sum_{i=1}^{I} d(p(i)).w(i)$$

$$= \min_{p(1),\ldots,p(I)} \left(d(p(1)).w(1) + \ldots + d(p(I)).w(I) \right)$$

$$= \min_{p(1),\ldots,p(I-1)} \left(d(p(1)).w(1) + \ldots + d(p(I-1)).w(I-1) \right)$$

$$+ \min_{p(I)} d(p(I)).w(I) \qquad (3.13)$$

Le théorème d'optimalité est ensuite appliqué à :

$$\underset{p(1),\ldots,p(I-1)}{\text{Min}} \quad (d(p(1)).w(1) + \ldots + d(p(I-1)).w(I-1)) \qquad (3.14)$$

et ainsi de suite.

Contraintes

Des contraintes sont imposées à la fonction d'alignement pour tenir compte des structures spécifiques de la parole. Nous rappelons ci-dessous les contraintes proposées par Sakoe-Chiba /78/, et que nous adopterons par la suite.

- Conditions aux limites :

$$\begin{aligned} m(1) &= n(1) = 1 \\ m(I) &= M \quad , \quad n(I) = N \end{aligned} \qquad (3.15)$$

- Conditions de monotonie et de continuité :

avec p(i) = (m(i),n(i)), on a pour p(i-1) les possibilités suivantes (Fig.3.2) :

$$p(i-1) = \begin{cases} (m(i) , n(i)-1) \\ \text{ou } (m(i)-1 , n(i)-1) \\ \text{ou } (m(i)-1 , n(i)) \end{cases} \qquad (3.16)$$

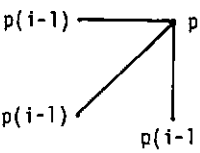


Fig.3.2

- Restriction de la pente du chemin d'alignement P. Cela revient à limiter le rapport des durées des deux portions du mot inconnu et du mot de la référence, mises en correspondance. Une pente comprise entre 1/2 et 2 est un choix raisonnable.

Les conditions de monotonie, de continuité et de pente impliquent des contraintes locales sur la fonction d'alignement P. Sakoe et Chiba /78/ proposent les contraintes locales de la figure 3.3.

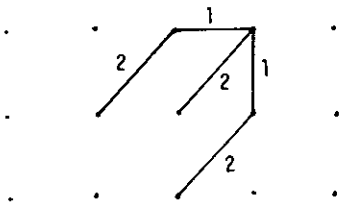


Fig.3.3 Contraintes locales selon Sakoe et Chiba

- $|m(i) - n(i)| \leq R$, $R > 0$ (3.17)

R définit une fenêtre d'ajustement temporelle tenant compte du fait que les fluctuations de durée et de rythme entre deux mots restent limitées.

D'autres variantes des contraintes peuvent être adoptées. Par exemple, Itakura /36/ propose les contraintes locales suivantes :

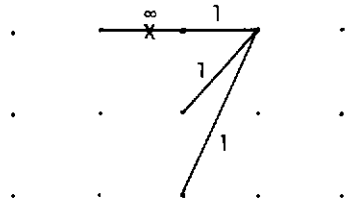


Fig.3.4 Contraintes locales selon Itakura

Okochi et Sakai proposent des contraintes locales qui rendent la fonction d'alignement temporel réversible dans le temps /69/.

Fonction de pondération w(i)

Plusieurs types de fonctions de pondération w(i) peuvent être utilisés. Le choix doit être effectué de telle façon que l'on puisse appliquer la technique de programmation dynamique. Pour cela, il suffit de rendre le dénominateur $\sum_i w(i)$ de la relation (3.12) indépendant de la fonction d'alignement P.

Il existe deux choix typiques /78/ :

1) forme symétrique

$$w(i) = (n(i)-n(i-1)) + (m(i)-m(i-1)) \qquad (3.18)$$

il suit :

$$\sum_i w(i) = N + M \qquad (3.19)$$

2) forme assymétrique :

$$w(i) = n(i) - n(i-1) \qquad (3.20)$$

il suit :

$$\sum_i w(i) = N \qquad (3.21)$$

Dans le cas d'une compression temporelle à la longueur constante N, $\sum_i w(i)$ est le même quels que soient les deux mots à comparer.

La distance dynamique entre A et B

En utilisant les contraintes de la figure 3.3 et une fonction de pondération selon (3.19), le calcul de la distance est déterminé récursivement selon la formule récursive suivante :

$$g(m,n) = d(m,n) + \text{Min} \begin{cases} g(m-1,n-2) + 2\ d(m,n-1) \\ g(m-1,n-1) + d(m,n) \\ g(m-2,n-1) + 2\ d(m-1,n) \end{cases} \qquad (3.22)$$

où g(m,n) est la distance cumulée au point (m,n). La distance entre A et B est alors :

$$D(A,B) = g(M,N) \ / \ (M+N) \qquad (3.23)$$

Dans tout ce qui suit, la distance dynamique entre deux mots sera calculée d'après (3.23).

3.4 Comparaison expérimentale des distances linéaires et dynamiques

Soit le test de reconnaissance monolocuteur TST A décrit dans le paragraphe 2.5.1.

Les figures 3.5a et 3.5b comparent les taux d'erreurs pour les trois façons de calculer la distance entre deux matrices : la distance simple (DMAT = 1), la distance linéaire avec déplacement global (DMAT = 2) et la distance dynamique (DMAT = 3). Il est à noter qu'après les différentes étapes du traitement numérique, les matrices contiennent 280 bits d'information.

Les résultats montrent clairement que la distance dynamique (DMAT=3) donne de meilleurs performances que les distances linéaires (DMAT=1 et DMAT=2); ils montrent aussi que la distance linéaire avec déplacement global (DMAT=2) est plus performante que la distance linéaire simple (DMAT=1).

Ces résultats montrent en outre que la normalisation curviligne (NTMP=2) contribue à faire le recalage temporel. Ceci est conforme aux résultats et interprétations donnés par d'autres auteurs /27/.

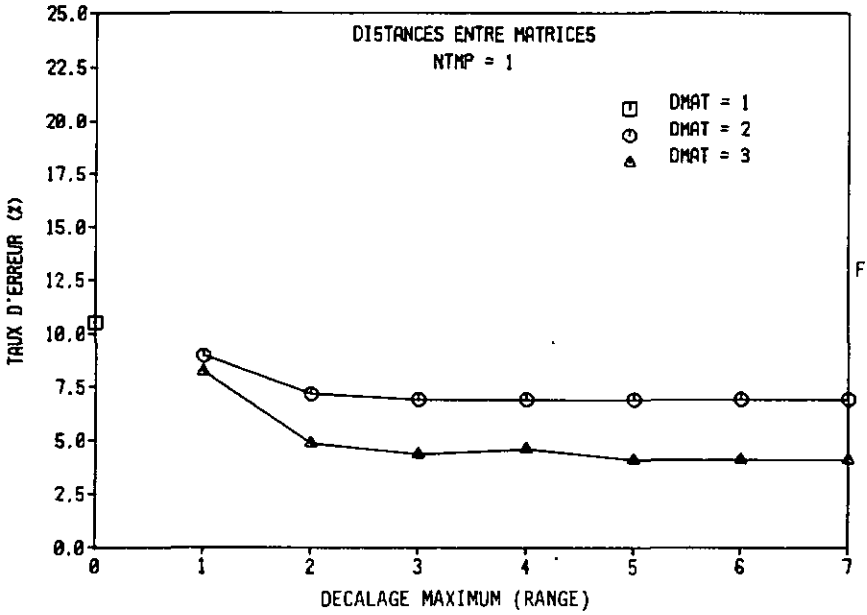


Fig.3.5a

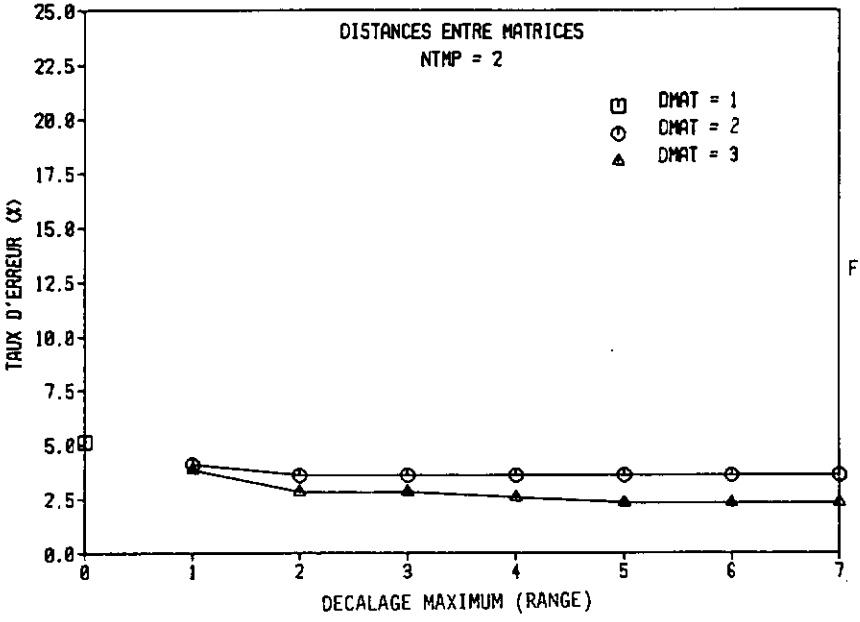


Fig.3.5b

Fig. 3.5 Comparaison des méthodes de calcul de distance entre matrices (en reconnaissance monolocuteur, TST A).

a) NTMP = 1

b) NTMP = 2

Choix des paramètres :

NTMP = 1,2 LN = 20

NAMP = Z2 LF = - LT = - C = 100

QUANT = 1 Nb = I rD = 0 r1 = 2

DMAT = Ø DVEC = 2 RANGE = Ø COF = 1

Remarque : D'autres tests ont montré que lorsque la distance entre deux vecteurs-fréquences est définie d'après (3.1), une pondération de la distance entre deux matrices binaires par l'inverse de la somme des "1" dans les deux matrices améliore légèrement les performances de reconnaissance par rapport au cas sans pondération.

3.5 A propos du rejet

Il y a rejet si la machine ne trouve pas de candidat dans son dictionnaire qui ressemble au mot inconnu.

Soit $D_k = D(\underline{A},\underline{R}_k)$ les distances obtenues entre le mot inconnu représenté par la matrice $\underline{A}$ et et les mots de référence représentés par les matrices $\underline{R}_k$, et telles que :

$$D_{k+1} \geq D_k \qquad \forall\, k = 1, 2, 3, \ldots, k, \ldots$$

C'est-à-dire R_1 est le plus proche de A, R_2 est le deuxième plus proche de A, etc.

Différentes stratégies de rejet peuvent être envisagées. En voici quelques-unes :

1) il y a rejet si :

$$D_1 > S$$

où S est un seuil de rejet unique déterminé empiriquement.

2) il y a rejet si la plus petite distance D_1 et la "seconde plus petite" D_2 sont très proches :

$$|D_1 - D_2| > S'$$

S' est également un seuil de rejet déterminé empiriquement (ici, on suppose qu'il y a une seule référence par mot).

3) il y a rejet si :

$$D_1 > S_1$$

où S_1 est le seuil de rejet associé au mot de la référence R_1. Ici, à chaque mot de la référence est associé un seuil de rejet différent.

3.6 Algorithme sans contrainte sur le début et la fin

La méthode d'alignement temporel dynamique décrite dans le paragraphe 3.3 force l'alignement des extrémités du mot inconnu avec ceux de la référence (relation 3.15). S'il y a des erreurs ou des imprécisions dans la détection du début ou de la fin de mot, l'alignement temporel ne sera pas effectué de façon optimale. Il en est de même dans le cas d'un bruit qui s'additionne au début ou à la fin du mot.

Exemple de problème :

- erreur courante de détection rencontrée avec des mots qui ont une plosive finale ('stop' qui devient 'sto')
- lorsque la prononciation commence ou finit par un petit claquement de lèvres

La méthode utilisée ici, consiste à étendre le domaine d'application de l'alignement temporel dynamique entre le mot inconnu et le mot de la référence au delà des extrémités. Cela revient à introduire un début et une fin virtuels. La figure 3.6 montre l'ensemble des couples de vecteurs qui participent à la détermination du chemin d'alignement.

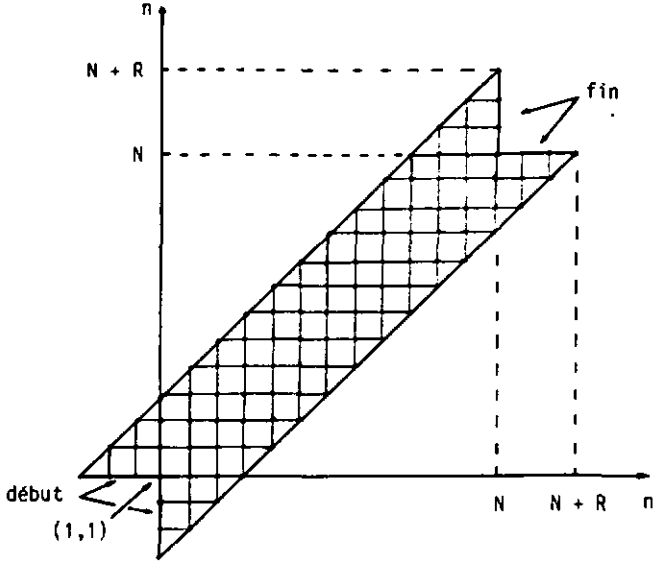


Fig.3.6 Alignement temporel dynamique sans contrainte sur le début et la fin des mots

Nous donnons dans la figure 3.7 une description symbolique de l'implémentation de l'algorithme de l'alignement dynamique avec début et fin libres. Les deux matrices $\underline{A}$ et $\underline{B}$ sont comprimées à une longueur fixe N. Les conditions suivantes sont imposées :

$$\underline{a}_n = \underline{0} \quad \text{et} \quad \underline{b}_n = \underline{0} \quad \forall \; n < 1 \text{ ou } n > N$$

Soient les définitions :

$$\begin{aligned}
G_n(r) &= g(n,n+r) && r = -R, \ldots, R \\
G_{n-1}(r) &= g(n-1,n-1+r) && r = -R, \ldots, R \\
G_{n-2}(r) &= g(n-2,n-2+r) && r = -R, \ldots, R \\
F_n(r) &= d(n,n+r) && r = -R, \ldots, R \\
F_{n-1}(r) &= d(n-1,n-1+r) && r = -R, \ldots, R
\end{aligned}$$

Où g(n,n) est défini par (3.22). Remarquons qu'en ce qui concerne la mémoire, on a besoin de 5 tableaux de dimension $2.(R+1) + 1$: G_n, G_{n-1}, G_{n-2}, F_n, F_{n-1}.

La figure 3.8 donne un exemple de chemin obtenu par l'algorithme de programmation dynamique avec la contrainte : les débuts et les fins des mots à comparer se correspondent respectivement. Les matrices de test représentent deux versions du mot 'quatre' prononcés par un même locuteur.

La figure 3.9 donne, dans le même cas, le chemin obtenu par l'algorithme de programmation dynamique sans contrainte sur les débuts et les fins des mots à comparer. L'avantage de l'algorithme sans contrainte sur les débuts et fins de mots apparaît clairement dans cette illustration.

Entrée : matrices $\underline{A}(K,N)$ et $\underline{B}(K,N)$, décalage maximum R
Sortie : distance $D(\underline{A},\underline{B})$

Initialisation :

$F_{n-1}(r) = 0 \qquad r = -R, \ldots, R$
$G_{n-2}(r) = 0 \qquad r = -R, \ldots, R$
$G_{n-1}(r) = 0 \qquad r = -R, \ldots, R$

$F_{n-1}(\pm(R+1)) = \infty$
$F_{n}(\pm(R+1)) = \infty$
$G_{n-2}(\pm(R+1)) = \infty$
$G_{n-1}(\pm(R+1)) = \infty$

Calcul de la distance :

DO for n = -R + 1 to N + R

DO for r = -R to R

Déterminer $F_n(r)$

$$G_n(r) = \text{Min} \begin{cases} G_{n-1}(r-1) + 2F_n(r-1) + F_n(r) \\ G_{n-1}(r) + 2F_n(r) \\ G_{n-2}(r+1) + 2F_{n-1}(r+1) + F_n(r) \end{cases}$$

ENDDO

DO for r = -R to R
$G_{n-2}(r) = G_{n-1}(r)$
$G_{n-1}(r) = G_n(r)$
$F_{n-1}(r) = F_n(r)$
ENDDO

ENDDO

$D(A,B) = \text{Min}_r\, G_n(r)$

Fig.3.7 Progamme de calcul de distance entre deux mots avec alignement temporel dynamique sans contrainte sur les début et fin de mots.

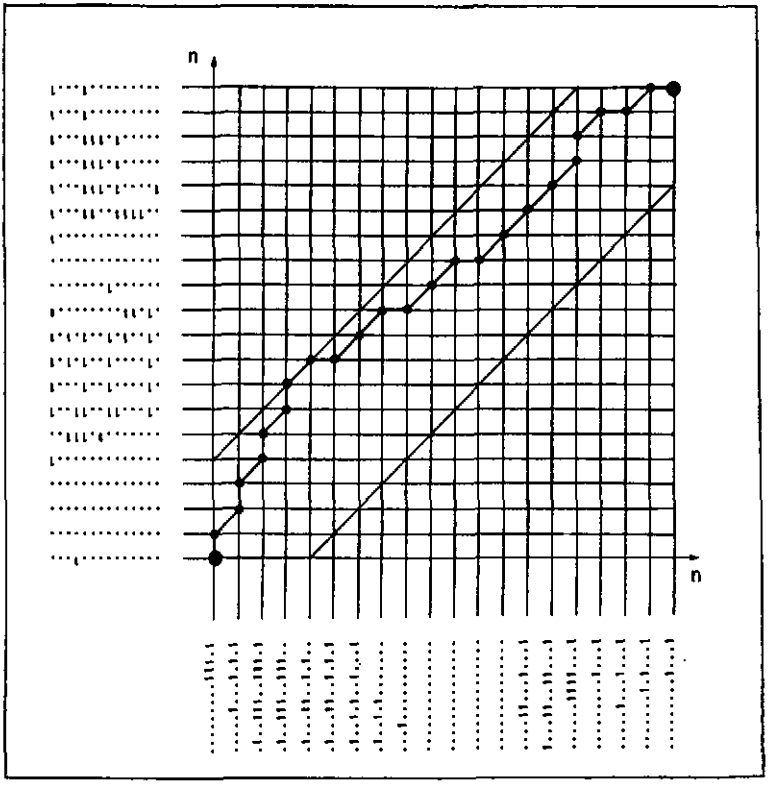


Fig.3.8 Exemple de chemin obtenu par l'algorithme de programmation dynamique avec la contrainte : les débuts et les fins des mots à comparer se correspondent respectivement. (Ici, on a deux versions du mot 'quatre' prononcés par un même locuteur).

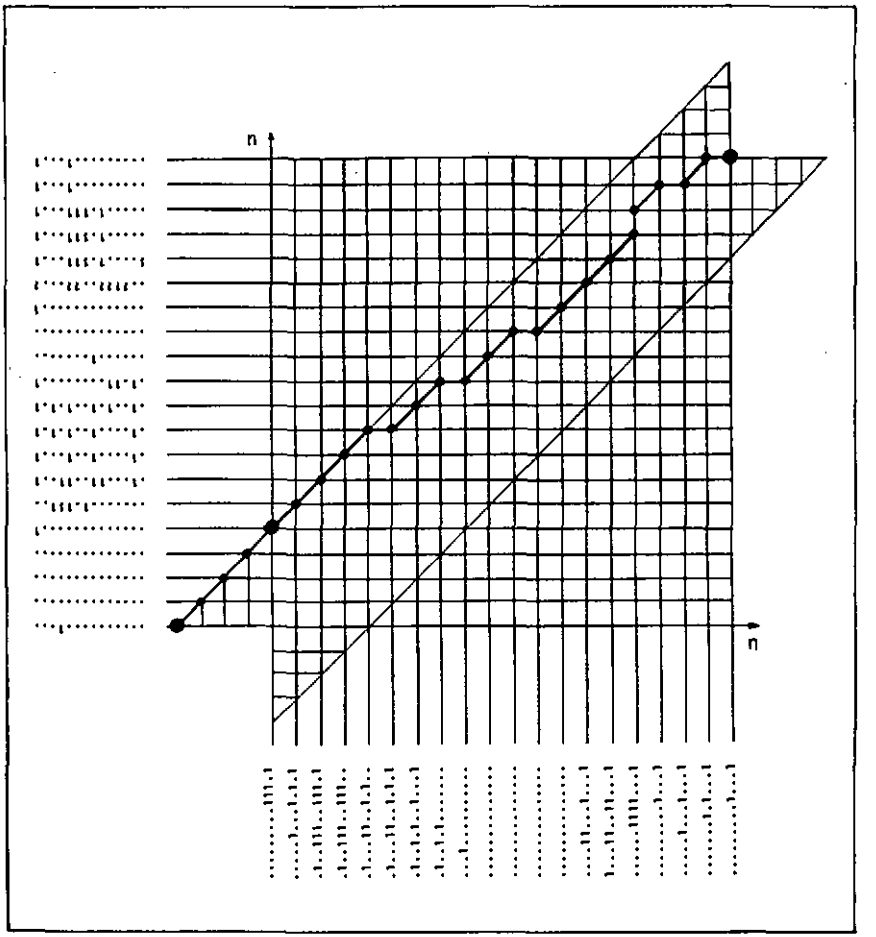


Fig.3.9 Exemple de chemin obtenu par l'algorithme de programmation dynamique sans contrainte sur les débuts et les fins des mots à comparer. (Même versions que dans Fig.3.8).

4. Reconnaissance multilocuteur par recalage fréquentiel

4.1 Motivation

Les méthodes de traitement vues jusqu'à maintenant s'avèrent insuffisantes dans un environnement multilocuteur. Il est nécessaire de tenir compte des différences entre locuteurs. Nous considérons ici le cas où l'on a une référence par mot.

Jusqu'à présent, les études effectuées pour trouver les paramètres qui différencient les individus sont essentiellement basées sur la fréquence fondamentale et la répartition des formants /49/.

On sait que la fréquence fondamentale d'une femme est environ le double de celle d'un homme. Les écarts par rapport à la valeur moyenne de la fréquence fondamentale sont très variables selon l'individu.

D'autre part, en ce qui concerne les formants on a pu établir, en première approximation, que les fréquences des formants (d'un son donné) d'un individu se déplacent de façon multiplicative par rapport à celles des formants d'un autre individu. Soit F1, F2, F3 les fréquences des trois premiers formants du premier individu, et F1', F2' , F3', celles du deuxième; on a :

$$F1 = a.F1' \ , \ F2 = b.F2' \ , \ F3 = c.F3'$$

avec a, b, c égaux et compris entre 0.8 et 1.2. Mais ceci est très approximatif, et il peut y avoir également des cas où de grandes différences entre a, b et c apparaissent /49/.

La répartition des largeurs de bande de nos filtres passes-bandes le long de l'axe fréquentiel est logarithmique : les deux premiers filtres couvrent chacun une octave, les suivants chacun un tiers d'octave.

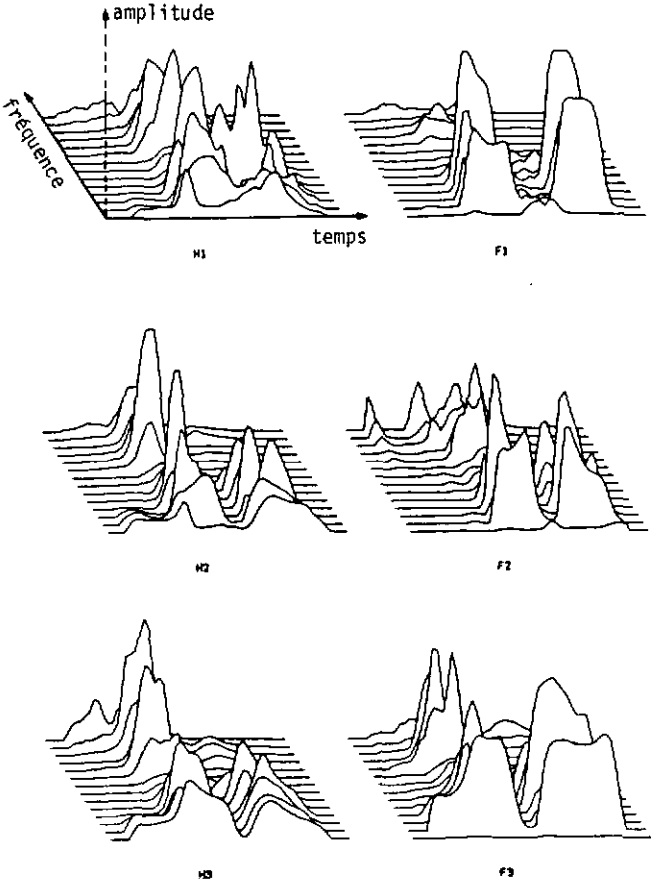


Fig.4.1 Spectrogrammes du mot "zéro" prononcé par trois hommes (H1, H2, H3) et trois femmes (F1, F2, F3).

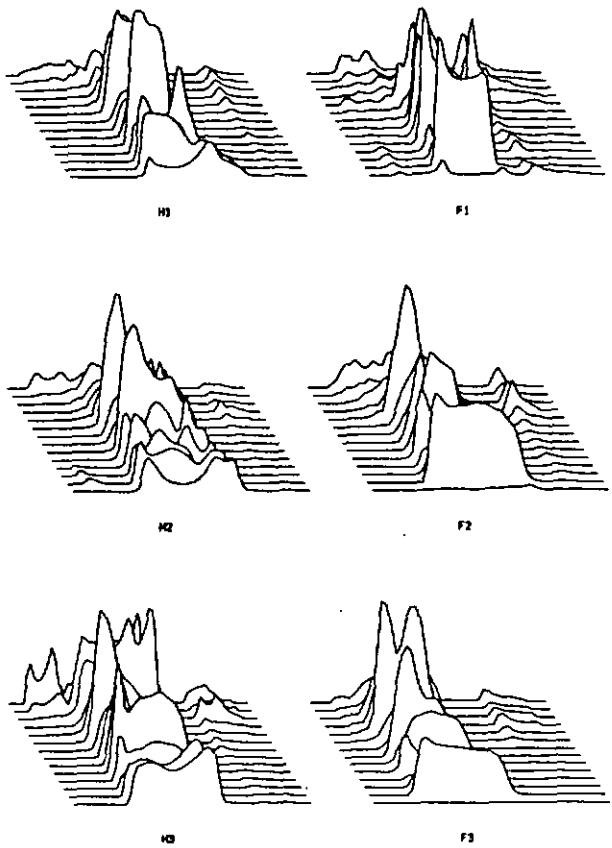


Fig.4.2 Spectrogrammes du mot "cinq" prononcé par trois hommes (H1, H2, H3) et trois femmes (F1, F2, F3).

La fréquence centrale du deuxième filtre est donc le double de celle du premier (environ 100 Hz). En admettant que la fréquence fondamentale de l'homme est d'environ 100 Hz, et celle de la femme d'environ 200 Hz, nous pouvons dire que la fréquence fondamentale d'un individu inconnu, homme ou femme, tombera dans le premier ou le deuxième canal (voir les exemples donnés dans les figures 4.1 et 4.2). Il y donc a un décalage possible d'un canal.

Pour les autres filtres, le rapport entre deux fréquences centrales successives est de 1.26. Une valeur de a comprise entre 0.8 et 1.2 a pour conséquence de faire tomber les fréquences des formants F1 et F1' dans deux canaux voisins. Comme pour la fréquence fondamentale, il y a donc aussi un décalage possible d'un canal. Il en est de même pour les deux autres formants.

Venons en aux méthodes de recalage fréquentiel proposées dans notre travail et testées.

La première méthode admet qu'il peut y avoir un décalage fréquentiel linéaire, constant en fonction du temps, d'un vecteur-fréquence par rapport à un autre; les deux vecteurs-fréquences appartiennent à deux prononciations d'un même mot par deux locuteurs différents.

La deuxième méthode admet qu'il peut y avoir un décalage fréquentiel linéaire, variable en fonction du temps, d'un vecteur-fréquence par rapport à un autre. Ceci tient compte du fait que, d'un individu à l'autre, il existe un décalage des vecteurs-fréquences dans le cas des sons voisés (par exemple canal 1 et 2), mais qu'il n'existe pas forcément de décalage pour les sons non-voisés.

La troisième méthode admet qu'il peut y avoir un décalage fréquentiel non-linéaire, variable en fonction du temps, d'un vecteur-fréquence par rapport à un autre.

4.2 Recalage fréquentiel linéaire constant en fonction du temps (RFLC)

Dans ce paragraphe, nous nous basons sur l'hypothèse que la variation qui peut exister entre deux locuteurs ou locutrices différents est dûe essentiellement à un décalage linéaire du vecteur-fréquence de l'un par rapport à l'autre (les deux vecteurs-fréquences appartenant à deux prononciations d'un même mot par deux locuteurs différents).

Les variations qui existent selon l'axe temporel sont traitées par

l'algorithme d'alignement temporel dynamique.

Le traitement proposé est donc la combinaison d'un alignement dynamique selon l'axe temporel et d'un recalage fréquentiel linéaire, constant en fonction du temps.

Reprenons la relation :

$$D(\underline{A},\underline{B}) = \min_{p} \sum_{i=1}^{I} d(p(i)).w(i)$$

qui décrit l'alignement temporel dynamique. Si on introduit le recalage fréquentiel, on obtient :

$$D(\underline{A},\underline{B}) = \min_{\delta} \min_{p} \sum_{i=1}^{I} \left(\sum_{k=1}^{K} d\left(a_{m(i)}(k) , b_{n(i)}(k+\delta) \right) . w(i) \right) \qquad (4.1)$$

avec : $\delta = -1 , 0 , +1$

Nous nous sommes limité dans le choix des valeurs de recalage à celles données ci-dessus. L'introduction de valeurs plus grandes seraient inutiles, car nous disposons de 14 canaux seulement.

4.3 Recalage fréquentiel linéaire variable en fonction du temps (RFLV)

La méthode de recalage fréquentiel du paragraphe précédent était basée sur l'hypothèse d'un décalage fréquentiel linéaire, constant en fonction du temps, d'un locuteur à l'autre. Ici, l'hypothèse est légèrement différente : le décalage est toujours linéaire selon l'axe fréquentiel, mais la valeur de décalage peut varier en fonction du temps. Ainsi :

$$D(\underline{A},\underline{B}) = \min_{p} \sum_{i=1}^{I} \min_{\delta} \left(\sum_{k=1}^{K} d\left(a_{m(i)}(k) , b_{n(i)}(k+\delta) \right) . w(i) \right) \qquad (4.2)$$

avec : $\delta = -1 , 0 , +1$

4.4 Recalage fréquentiel dynamique (RFD)

Dans ce paragraphe, en plus de l'alignement dynamique selon l'axe temporel, nous utilisons également la programmation dynamique pour aligner les vecteurs-fréquences, c'est-à-dire qu'on effectue également un alignement dynamique selon l'axe fréquentiel.

Donc, on a :

$$D(\underline{A},\underline{B}) = \min_{P} \sum_{i=1}^{I} d(p(i)).w(i)$$

avec :

$$d(p(i)) = \min_{P'} \sum_{i'=1}^{I'} d(p'(i')).w'(i') \qquad (4.3)$$

Soient 2 vecteurs-fréquences représentés par leurs K valeurs binaires correspondant aux K canaux. Par exemple :

$\underline{a}_m$ = 10001000000011

$\underline{b}_n$ = 01001000000111

L'introduction du recalage dynamique entre ces deux vecteurs-fréquences devrait donner une distance $d(p(i)) = d(\underline{a}_m,\underline{b}_m)$ nulle.

Soient les deux vecteurs-fréquences $\underline{a}_m$ et $\underline{b}_n$ représentés par leurs composantes:

$\underline{a}_m$ =(a(1) a(2) ... a(k) ... a(K))
$\underline{b}_n$ =(b(1) b(2) ... b(l) ... b(K))

La fonction de recalage F' entre ces deux vecteurs est exprimée par la séquence des couples :

$$p'(i') = (a(k(i')) , b(j(i'))) \qquad (4.4)$$

c'est-à-dire :

$$P' = p'(1), p'(2), ...p'(I') \qquad (4.5)$$

La distance d(p'(i')) entre a(k(i')) et b(j(i')) est par exemple :

$$d(p'(i')) = |a(k(i')) - b(j(i'))|$$

Les contraintes utilisées lors du recalage dynamique selon l'axe fréquentiel sont les mêmes que celles utilisées selon l'axe temporel, sauf la largeur de la fenêtre qui est plus petite. Nous utilisons par ailleurs l'algorithme sans contraintes sur le début et la fin des mots pour la détermination du chemin F'.

4.5 Evaluation expérimentale

Dans les tests qui suivent, nous évaluons les performances en reconnaissance multilocuteur des différentes méthodes de recalage fréquentiel que nous venons de décrire.
Les performances en reconnaissance monolocuteur sont également mesurées.

Rappelons qu'en reconnaissance monolocuteur (R.mono), le locuteur-référence est le même que le locuteur-test; par contre en reconnaissance multilocuteur (R.multi), le locuteur-référence est différent du locuteur-test.

Soit les 4 cas suivants :

Cas SRF : reconnaissance Sans Recalage Fréquentiel
(même méthode qu'en reconnaissance monolocuteur)

Cas RFLC : reconnaissance avec Recalage Fréquentiel Linéaire Constant
(paragraphe 4.2)

Cas RFLV : reconnaissance avec Recalage Fréquentiel Linéaire Variable
(paragraphe 4.3)

Cas RFD : reconnaissance avec Recalage Fréquentiel Dynamique
(paragraphe 4.4)

Dans tous les cas nous utilisons l'alignement temporel dynamique.

Conditions des tests :

TST C : Les tests de reconnaissance sont effectués sur la base du même vocabulaire décrit dans §2.5.
6 locuteurs (3 hommes : H1, H2, H3 et 3 femmes : F1, F2, F3) participent aux tests de reconnaissance. 5 versions du vocabulaire par locuteur sont utilisées, les trois dernières servent à créer la référence de chaque locuteur.

Les tests de reconnaissance sont effectués avec les paramètres de traitement suivants (voir annexe A):

NAMP = Z2
NTMP = 1 N = 2D
QUANT = 1 Nb = 1 rD = 0 r1 = 2
DMAT = 3 DVEC = 1 RANGE = 4 CDF = D

Les paramètres du recalage fréquentiel sont choisis comme suit :

Cas RFLC : δ = -1, 0, +1
Cas RFLV : δ = -1, 0, +1
Cas RFD : * Largeur de la fenêtre pour la comparaison des vecteurs-fréquences égale à 2.
* Début et fin libres selon l'axe fréquentiel

Résultats

Les tableaux de la figure 4.3 donnent les taux d'erreur de reconnaissance (en %) pour les quatre cas indiqués plus haut (SRF, RFLC, RFLV et RFD) dans un environnement multilocuteur (R.multi) et monolocuteur (R.mono).

Des résultats obtenus, il sort les faits suivants :

1) très faible détérioration de la reconnaissance monolocuteur lorsqu'on effectue un recalage fréquentiel

2) peu de différences entre les performances obtenues par les méthodes de recalage fréquentiel RFLC, RFLV et RFD

3) Globalement la reconnaissance multilocuteur passe d'un taux d'erreur de 25.5 % sans recalage fréquentiel à 16 % avec recalage fréquentiel.

4.6 Conclusion

L'objectif du recalage fréquentiel était le suivant : sur la base d'une référence créée avec un seul locuteur, faire une reconnaissance multilocuteur en effectuant selon l'axe temporel un alignement dynamique et selon l'axe fréquentiel un recalage RFLC, RFLV ou RFD.

Les résultats montrent que le recalage fréquentiel produit une diminution substantielle du taux d'erreur lors d'essai de reconnaissance inter-genre (le locuteur-référence est un homme et le locuteur-test une femme, ou vice-versa). Par contre, aucune amélioration significative n'est obtenue dans le cas de la reconnaissance intra-genre.

Ainsi, le recalage fréquentiel pourrait servir à compenser les différences inter-genre. Toutefois, cette méthode ne pourrait se suffire à elle même, car les taux d'erreur restent relativement élevés. Par contre, elle pourrait compléter d'autres méthodes exposées dans les chapitres suivants.

Cas SRF :

R.multi

	H	F
H	6.5	40.56
F	40.11	15.17

R.mono

H	F
0.	0.

Cas RFLC :

R.multi

	H	F
H	7.8	19.0
F	23.0	15.2

R.mono

H	F
0.0	0.0

Cas RFLV :

R.multi

	H	F
H	8.7	14.7
F	22.7	18.

R.mono

H	F
0.0	1.3

Cas RFO :

R.multi

	H	F
H	7.3	18.1
F	25.3	17.2

R.mono

H	F
0.0	1.3

Fig.4.3 Resultats des tests de reconnaissance sans et avec recalage fréquentiel

5. Adaptation automatique au locuteur

L'adaptation automatique au locuteur consiste à changer les références obtenues lors de l'apprentissage d'un locuteur quelconque en des références valables pour un nouveau locuteur sans apprentissage de ce dernier.

Deux approches peuvent être envisagées : l'adaptation par transformation et l'auto-adaptation par détection automatique d'erreurs.

5.1 Adaptation au locuteur par transformation

Le principe de l'adaptation par transformation est le suivant. Soit R_m, m = 1,...,M l'ensemble des références pour les M mots prononcés par un locuteur L.
L'adaptation à un nouveau locuteur L' consiste à déterminer une transformation T qui permet de trouver, à partir des références R_m du locuteur L, des nouvelles références R'_m valables pour le locuteur L' :

$$R_m \xrightarrow{T} R'_m$$

La transformation T doit être déterminée sur la base des prononciations faites par les locuteurs L et L' d'une seule locution type. Bien que séduisante, cette approche est limitée par la difficulté de choisir une transformation adéquate. Pour plus de détails voir /26/.

5.2 Auto-adaptation au locuteur par détection automatique d'erreurs

Cette deuxième approche est plus intéressante d'un point de vue pratique /9/.

Au départ, on dispose dans le système de reconnaissance d'un ensemble de M mots de référence obtenu lors de l'apprentissage d'un locuteur quelconque.

On suppose que l'utilisateur du système de reconnaissance dispose d'un moyen qui lui permet de contrôler si le système a reconnu correctement le mot ou non; par exemple : affichage du mot reconnu, réponse vocale par synthèse,

etc.

Le principe est le suivant :

Toute répétition d'un mot par le locuteur est interprétée par le système de reconnaissance comme une erreur de reconnaissance (sauf dans certains cas) qu'il faut corriger.

Si la réponse donnée par le système est effectivement fausse, il suffit à l'utilisateur de répéter le mot qui a été mal reconnu, jusqu'à ce que le système donne la bonne réponse (nous verrons plus loin comment le système arrive à donner la bonne répone).

Dès que la réponse du système et le mot prononcé concordent, celui-ci est pris par le système comme nouvelle référence pour ce mot (nous verrons plus loin comment le système arrive à savoir que mot prononcé et réponse concordent).

Nous allons détailler la procédure d'auto-adaptation en considérant d'abord que le système "conclut" à l'erreur dès qu'une répétition est détectée (la répétition n'est alors pas permise s'il n'y a pas d'erreur de reconnaissance).

Soit

- A le mot d'entrée actuel (il s'agit du dernier mot prononcé),
- A_0 le mot d'entrée précédent,
- $D(A,R_m)$ la distance entre le mot d'entrée actuel A et le mot de référence R_m,
- $D(A,A_0)$ la distance entre le mot d'entrée actuel A et le mot d'entrée précédent A_0
- i(n) le "n-ième plus proche" mot de la référence de A :

$$D(A,R_{i(1)}) \leq D(A,R_{i(2)}) \leq D(A,R_{i(3)}) \quad .... \qquad (5.1)$$

- $i_0(n)$ le "n-ième plus proche" mot de la référence de A_0 :

$$D(A_0,R_{i_0(1)}) \leq D(A_0,R_{i_0(2)}) \leq D(A_0,R_{i_0(3)}) \quad .... \qquad (5.2)$$

Chaque fois que l'on prononce un nouveau mot A, son précédent devient le mot A_0.

Le système considère qu'il y a répétition si A est plus proche de son précédent A_0 que des références R_m. C'est-à-dire si la relation suivante est vérifiée :

$$D(A,A_0) < Dmin \qquad (5.3)$$

avec :

$$Dmin = \min_m D(A,R_m) \qquad (5.4)$$

Dans ce cas (5.3 vérifiée), le système ne donne pas la réponse $R_{i(1)}$ qui correspond au mot de la référence le plus proche du mot d'entrée actuel A, mais $R_{i_0(2)}$ qui est le précédent "deuxième plus proche" (c'est-à-dire le "deuxième plus proche" mot de la référence de A_0). S'il y a encore une répétition le sytème donnera comme réponse $R_{i_0(3)}$, et ainsi de suite. Lorsque le mot affiché R* (par exemple $R_{i_0(2)}$) correspond au mot prononcé A*, le locuteur prononce n'importe quel autre mot du vocabulaire. Le système détecte alors cette non-répétition et remplace la référence R* par A*. Puis, il donne la réponse au dernier mot d'entrée.

Il est important de noter que la liste $i_0(n)$ est mise à jour, si et seulement si, le mot prononcé est reconnu correctement :

$$i_0(n) = i(n) \text{ pour } n = 1,\ldots,M \qquad \text{(s'il n' y pas de répétition)}$$

La figure 5.1 donne une illustration du déroulement de la procédure. Les tests effectués avec des petits vocabulaires montrent qu'après une erreur de reconnaissance, la bonne réponse est obtenue en général après une ou deux répétitions .

Jusqu'à présent nous avons considéré qu'une répétition d'un mot sous-entendait automatiquement qu'il a été mal reconnu. On ne peut donc prononcer deux mots identiques à la suite. Dans ce qui suit,on va lever cette contrainte, pour faire en sorte que le système de reconnaissance ne "conclue" pas forcément à l'erreur s'il détecte une répétition.

Nous pouvons considérer que le système a deux états :

1) "l'état normal"
2) "l'état de correction d'erreur et d'auto-apprentissage".

Le système ne considère pas la répétition comme une erreur, si les conditions suivantes sont vérifiées :

a) le système est dans l'état normal
b) le mot reconnu actuellement est le même que le mot reconnu précédemment ($R_{i(1)} = R_{i_q(1)}$).
c) la distance Dmin est inférieure à un seuil S.

Les tests que nous avons effectués avec l'introduction de trois conditions a), b) et c) ont montré clairement que le système de reconnaissance interprète une répétition comme une erreur, seulement s'il y a réellement une erreur de reconnaissance.

La description complète de l'algorithme est donnée par la figure 5.2

Dans l'algorithme tel qu'il est décrit dans la fig.5.2, on ne sort de "l'état correction d'erreur et auto-apprentissage" qu'en prononçant un mot différent du 'mot répété'. On peut remplacer cette exigence en adoptant une autre variante de cet algorithme : après un certain délai (par exemple 4 secondes) le système considère que la réponse (mot affiché) qu'il vient de donner est juste /9/.

5.3 Discussion

L'algorithme de l'auto-adaptation au locuteur permet de changer les mots de référence pour les adapter à un nouveau locuteur pendant l'utilisation du système sans apprentissage apparent. Il permet également de suivre les modifications à court terme de la voix d'un locuteur.

Cette méthode exige un contrôle permanent du mot reconnu par l'utilisateur (il y a une contre-réaction de l'utilisateur sur le système de reconnaissance). Seuls les systèmes portables conviennent à cette exigence.

L'analyse et le test de cette méthode sur ordinateur ont montré qu'elle est surtout adaptée pour des petits vocabulaires, ne dépassant pas par exemple une dizaine de mots.

Cette méthode peut aussi être combinée avec un système multilocuteur, où les références sont déjà prévues pour un large public (voir prochains chapitres)

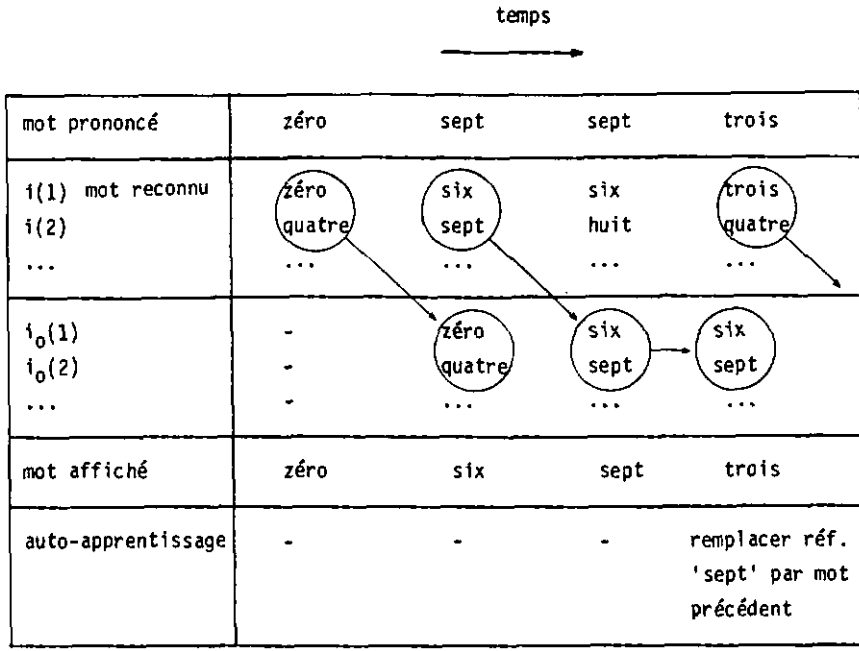


Fig.5.1 Illustration de la procédure d'auto-adaptation

Notation pour la figure 5.2 :

$\underline{A}$: Matrice correspondant au mot d'entrée actuel
$\underline{A}_o$: Matrice correspondant au mot d'entrée précédent
$\underline{R}_m$: Matrice correspondant au m-ième mot de la référence
$D(\underline{A},\underline{R}_m)$: Distance entre les matrices $\underline{A}$ et $\underline{R}_m$
Dmin : Distance minimale entre A et les R_m
i(n) : "n-ième plus proche" mot de la référence de A
$i_o(n)$: "n-ième plus proche" mot de la référence de A_o
S : Seuil de distance

1. PARLER : A

2. Calculer a) $D_{min} = \min_{m} D(\underline{A},\underline{R}_m)$

   b) $i(n)$, $n = 1,\ldots,M$

3. Si $D_{min} \leq D(\underline{A},\underline{A}_0)$ aller à 13.

4. Si $i(1) = i_o(1)$ et Si $D_{min} < S$ aller à 13.

5. $j = 1$

6. $j = j + 1$

7. REPONSE : $i_o(j)$

8. $\underline{A}_0 = \underline{A}$

9. PARLER : A

10. Calculer a) $D_{min} = \min_{m} D(\underline{A},\underline{R}_m)$

    b) $i(n)$, $n = 1,\ldots,M$

11. Si $D_{min} > D(\underline{A},\underline{A}_0)$ aller à 6.

12. $\underline{R}_{i_o(j)} = \underline{A}_0$

Correction d'erreur et auto-apprentissage

13. $\underline{A}_0 = \underline{A}$

    $i_o(n) = i(n)$, $n=1,\ldots,M$

14. REPONSE : i(1) et aller à 1.

Fig.5.2 Algorithme d'auto-adaptation au locuteur

6. Analyse factorielle appliquée aux échantillons multilocuteurs de la parole

6.1 Introduction

La reconnaissance multilocuteur de la parole peut être envisagée par la méthode des références multiples. Dans cette dernière, on cherche à décrire, par un nombre limité de classes, une même locution prononcée par une large population. Les méthodes de classification automatique traitées dans le prochain chapitre permettent de générer ces classes. Avant de chercher des classes au sein d'un ensemble de prononciations, on fait implicitement l'hypothèse que ces classes existent.

Le but poursuivi ici est de représenter et d'analyser la répartition des prononciations d'un (ou plusieurs) mot(s) par plusieurs locuteurs pour voir si l'hypothèse de l'existence de ces classes se justifie. L'analyse est effectuée visuellement. Une prononciation, appelée échantillon par la suite, est caractérisée par un grand nombre de composantes, en l'occurence les éléments de la matrice-spectrogramme.

La visualisation dans l'espace défini par l'ensemble des composantes n'étant pas possible, on cherche à obtenir une représentation dans un sous-espace telle que l'information la plus essentielle sur la répartition de ces échantillons ne soit pas perdue. Si par exemple, la dimension du sous-espace retenu est 2, une simple inspection visuelle apporte des renseignements concernant la séparation entre d'éventuels groupes d'échantillons, le repérage d'échantillons isolés, etc.

Soit le nuage des I échantillons représentés dans l'espace des composantes. Le problème consiste à trouver un sous-espace ayant un petit nombre de dimensions (par exemple le plan) dans lequel on puisse faire une représentation des échantillons sans trop modifier les distances initiales entre les échantillons. En projetant les échantillons sur ce sous-espace,

les distances entre les échantillons projetés sont inférieures aux distances initiales. Le meilleur sous-espace est celui pour lequel les distances entre les projections sont les plus proches possibles des distances initiales.

Ainsi, l'objectif principal de l'analyse factorielle est de résumer l'information apportée par un grand nombre de composantes, par un petit nombre plus restreint de nouvelles composantes. Les nouvelles composantes obtenues ne sont pas une simple sélection parmi les composantes de départ : ce sont de nouvelles composantes, appelées composantes principales, réalisant la synthèse de plusieurs composantes initiales /8/,/10/,/15/,/46/.

6.2 Formulation du problème de visualisation des échantillons de parole

Un échantillon (prononciation d'un mot par un locuteur) est caractérisé en dernier lieu par une matrice. Nous pouvons alors calculer la distance entre deux échantillons comme étant la distance dynamique (voir chapitre 3.) entre les deux matrices correspondantes. Soient I échantillons obtenus par la prononciation d'un (ou plusieurs) mot(s) par I locuteurs. La distance D(i,j) entre les échantillons i et j est donc connue pour tout i et j.

Sous certaines conditions /86/, on peut représenter les I échantillons dans un espace euclidien E^q de dimension q ($q \leq I-1$) de telle façon que les distances entre les échantillons soient exactement respectées. Chaque échantillon i sera représenté dans l'espace E^q par un vecteur $\underline{x}_i$.

Une représentation dans l'espace E^q n'est évidemment ni praticable ni propice à une analyse visuelle. On cherche alors une représentation de tous les échantillons dans un sous-espace F^m dont la dimension m est beaucoup plus petite que q (pour une analyse visuelle, m doit être plus petit ou égal à 3). La distance D'(i,j) dans le sous-espace F^m doit être la plus proche possible de D(i,j).

La formulation du problème est donc la suivante : étant donné I échantillons et les distances D(i,j) entre-eux, il s'agit de les représenter par I vecteurs $\underline{x}_1$, $\underline{x}_2$, ..., $\underline{x}_I$ de F^m, par exemple m = 2, de telle façon que les distances D'(i,j) dans F^m soient les plus proches possibles des distances réelles D(i,j). Notons bien que l'on cherche une représentation des échantillons à partir de leurs distances mutuelles.

6.3 Préliminaires /15/

Dans ce paragraphe, on rappelle brièvement quelques aspects mathématiques concernant l'analyse en composantes principales qui est considérée comme la méthode de base des méthodes factorielles.

Soit $\underline{x}_i$, i = 1, ..., I l'ensemble des vecteurs dans un espace E^q de dimension q. Le vecteur $\underline{x}_i$ contient les q composantes mesurées pour l'échantillon i. Une observation du nuage dans l'espace E^q est impraticable si q est supérieure à 3. On cherche alors un sous-espace de projection de dimension inférieure (par exemple le plan) de telle façon que la déformation du nuage en projection soit minimale.

Soit $\underline{g}$ le centre de gravité défini par :

$$\underline{g} = \frac{1}{I} \sum_{i}^{I} \underline{x}_i \qquad (6.1)$$

Faisons un changement d'origine en $\underline{g}$. Chaque échantillon sera alors représenté par :

$$\underline{y}_i = \underline{x}_i - \underline{g} \qquad (6.2)$$

Soit $\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_m, \ldots, \underline{v}_q$ une base orthonormée de E^q, F^m le sous-espace engendré par $\underline{v}_1, \ldots, \underline{v}_m$ et $F^{m'}$ le sous-espace engendré par $\underline{v}_{m+1}, \ldots, \underline{v}_q$. F^m et $F^{m'}$ étant orthogonaux on a pour tout i :

$$|\underline{y}_i|^2 = \sum_{k=1}^{q} |<\underline{y}_i, \underline{v}_k>|^2 = |<\underline{y}_i, F^m>|^2 + |<\underline{y}_i, F^{m'}>|^2 \qquad (6.3)$$

$<\underline{y}_i, \underline{v}_k>$ est la projection de $\underline{y}_i$ sur $\underline{v}_k$, $<\underline{y}_i, F^m>$ est la projection de $\underline{y}_i$ sur F^m. $<\underline{y}_i, F^{m'}>$ est la projection de $\underline{y}_i$ sur $F^{m'}$.

En sommant sur i, on a :

$$\sum_{i} |\underline{y}_i|^2 = \sum_{i} |<\underline{y}_i, F^m>|^2 + \sum_{i} |<\underline{y}_i, F^{m'}>|^2 \qquad (6.4)$$

Le membre de gauche de la relation (6.4) est une caractéristique de l'ensemble des échantillons et ne dépend pas du choix des sous-espaces F^m et

$F^{m'}$. Comme l'origine des vecteurs $\underline{y}_i$ se confond avec leur centre de gravité, le membre de gauche de la relation (6.4) est égal, par définition, à l'inertie du nuage par rapport au centre de gravité $\underline{g}$. Il est aussi directement proportionnel à la somme des carrés des distances entre les échantillons /10/.

Comme l'opération de projection raccourcit toujours les distances, il faut choisir le sous-espace de projection F^m dans lequel les distances seront en moyenne les plus grandes (c'est-à-dire les mieux conservées). On se fixe alors comme critère de rendre maximale la somme des carrés des distances entre les projections. Ce critère est équivalent à rendre maximale l'inertie du nuage projeté par rapport à $\underline{g}$.

Donc, on doit rendre la somme

$$\sum_i |<\underline{y}_i, F^m>|^2 \qquad (6.5)$$

maximale. En d'autre termes, il s'agit de trouver $\underline{v}_1$, $\underline{v}_2$, ..., $\underline{v}_m$ tels que

$$\sum_{k=1}^{m} \sum_{i=1}^{I} |<\underline{y}_i, \underline{v}_k>|^2 \quad \text{est maximum} \qquad (6.6)$$

Soit $\underline{V}_k$ la matrice unicolonne contenant les q composantes de $\underline{v}_k$ et $\underline{Y}$ la matrice à I lignes et q colonnes dans laquelle la i-ème ligne contient les q composantes du vecteur $\underline{y}_i$

Le terme $\sum_i |<\underline{y}_i, \underline{v}_k>|^2$ peut être écrit sous la forme matricielle suivante :

$$\underline{V}'_k . \underline{Y}' . \underline{Y} . \underline{V}_k \qquad (6.7)$$

$\underline{Y}'$ et $\underline{V}'$ sont les transposées respectives de $\underline{Y}$ et $\underline{V}$.

Donc (6.6) équivaut à

$$\text{maximiser} \quad \sum_{k=1}^{m} \underline{V}'_k . \underline{Y}' . \underline{Y} . \underline{V}_k$$

$$\underline{V}_k \text{ sont les inconnus avec } \underline{V}'_i . \underline{V}_j = \begin{cases} 1 & \text{si } i=j \\ 0 & \text{si } i \neq j \end{cases} \qquad (6.8)$$

La solution du problème (6.8) est bien connue /16/,/46/ :
si $\lambda_1, \lambda_2, \ldots, \lambda_m, \ldots, \lambda_q$ sont les valeurs propres par ordre décroissant de $\underline{Y}'.\underline{Y}$ et $\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_m, \ldots, \underline{v}_q$ les vecteurs propres associés, alors les m premiers vecteurs propres normés constituent la solution du problème.

Ainsi, le meilleur sous-espace de dimension m pour la représentation des I échantillons est celui formé par les m vecteurs propres $\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_m$.

La composante de l'échantillon i sur le vecteur $\underline{v}_k$ est :

$$< \underline{y}_i , \underline{v}_k > \qquad (6.9)$$

Propriétés

- Si $\underline{Y}$ est symétrique ($\underline{Y} = \underline{Y}'$) :

$$\underline{Y} . \underline{V}_k = \sqrt{\lambda}_k . \underline{V}_k \qquad (6.10)$$

- $$\forall k \quad \sum_{i=1}^{I} | < \underline{y}_i , \underline{v}_k > |^2 = \lambda_k \qquad (6.11)$$

- $$\text{trace } (\underline{Y}'\underline{Y}) = \sum_{i=1}^{I} \underline{y}_i = \sum_{k=1}^{q} \lambda_k \qquad (6.12)$$

(trace : somme des éléments de la diagonale)

La relation (6.11) mesure l'inertie du nuage d'échantillons projeté sur l'axe factoriel $\underline{v}_k$. La relation (6.12) mesure l'inertie du nuage d'échantillons dans E^q. Ces deux dernières relations permettent, suivant le nombre d'axes factoriels choisi (le nombre m de $\underline{v}_k$), de mesurer la quantité d'information conservée par le facteur t_m suivant :

$$t_m = \frac{\lambda_1 + \lambda_2 + \ldots + \lambda_m}{\lambda_1 + \lambda_2 + \ldots + \lambda_q} \qquad (0 < t_m \leq 1) \qquad (6.13)$$

Remarquons que si pour un m donné, $t_m = 1$, il n' y a aucune perte d'information en passant de E^q à F^m.

6.4 Analyse des échantillons multilocuteurs d'après les distances mutuelles

Les échantillons multilocuteurs sont l'ensemble des prononciations d'un (ou plusieurs) mot(s) par plusieurs locuteurs. Comme cela a été indiqué précédemment, l'information de départ est donnée par les distances entre les échantillons multilocuteurs.

Supposons que les distances vérifient les propriétés suivantes :

$$\begin{aligned} &(1)\ D(i,i) = 0 && \forall\ i \\ &(2)\ D(i,j) = D(j,i) && \forall\ i \text{ et } j \qquad (6.14) \\ &(3)\ D(i,k) \leq D(i,j) + D(j,k) && \forall\ i \neq j \neq k \end{aligned}$$

Dans notre cas, les deux premières propriétés sont toujours vérifiées d'après la définition même de la distance choisie (voir 3.23). Par contre, la troisième ne l'est pas forcément pour tous les triplets i, j et k. En pratique, on peut l'admettre. En effet, en observant un très grand nombre de distances entre divers prononciations, on n'a relevé aucun cas qui contredit la troisième propriété de (6.14).

Dans le cas où les trois propriétés précédentes sont vérifiées, il est possible qu'il existe un entier $q \leq I-1$ et I vecteurs $\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_I$ appartenant à E^q tels que

$$D(i,j)^2 = (\underline{x}_i - \underline{x}_j)^2 \quad \forall\ i \text{ et } j \qquad (6.15)$$

Précisons que la vérification des propriétés de la relation (6.14) est une condition nécessaire mais pas suffisante pour trouver I vecteurs appartenant à E^q vérifiant (6.15). Nous verrons plus loin la condition nécessaire et suffisante.

Passage du tableau de distance à $\underline{Y}'.\underline{Y}$

Supposons que l'origine est au centre de gravité des $\underline{x}_i$. On va montrer que le produit scalaire entre deux vecteurs centrés $\underline{y}_i$ et $\underline{y}_j$ (c'est-à-dire le terme se trouvant sur la i-ème ligne et la j-ème colonne de $\underline{Y}'.\underline{Y}$) ne dépend que des distances euclidiennes entre les échantillons. Donc, on peut déterminer la matrice $\underline{Y}'.\underline{Y}$ en connaissant uniquement les distances entre échantillons.

$$D(i,j)^2 = D(\underline{x}_i,\underline{x}_j)^2 = (\underline{x}_i - \underline{x}_j)^2 = \underline{x}_i^2 + \underline{x}_j^2 - 2.\underline{x}_i.\underline{x}_j$$

d'où :

$$\underline{x}_i.\underline{x}_j = -\frac{1}{2}\{ D(i,j)^2 - \underline{x}_i^2 - \underline{x}_j^2 \} \qquad (6.15')$$

Il s'agit de trouver le produit $\underline{y}_i.\underline{y}_j$:

$$\underline{y}_i.\underline{y}_j = (\underline{x}_i - \underline{g}).(\underline{x}_j - \underline{g})$$

En utilisant (6.1) et (6.15'), on a :

$$\underline{y}_i.\underline{y}_j = (\underline{x}_i - \frac{1}{I}\sum_{k=1}^{I} \underline{x}_k) \cdot (\underline{x}_j - \frac{1}{I}\sum_{k=1}^{I} \underline{x}_k)$$

$$= \underline{x}_i.\underline{x}_j - \frac{1}{I}\sum_k \underline{x}_i.\underline{x}_k - \frac{1}{I}\sum_k \underline{x}_k.\underline{x}_j + \frac{1}{I^2}\sum_k\sum_l \underline{x}_k.\underline{x}_l$$

$$= \frac{1}{2}\{ - D(i,j)^2 + \underline{x}_i^2 + \underline{x}_j^2$$

$$+ \frac{1}{I}\sum_k D(i,k)^2 - \underline{x}_i^2 - \frac{1}{I}\sum_k \underline{x}_k^2$$

$$+ \frac{1}{I}\sum_k D(k,j)^2 - \frac{1}{I}\sum_k \underline{x}_k^2 - \underline{x}_j^2$$

$$- \frac{1}{I^2}\sum_k\sum_l D(k,l)^2 + \frac{1}{I^2}\sum_k\sum_l (\underline{x}_k^2 + \underline{x}_l^2) \}$$

Finalement, on obtient :

$$\underline{y}_i.\underline{y}_j = -\frac{1}{2} D(i,j)^2 + \frac{1}{2I}\sum_k D(i,k)^2 + \frac{1}{2I}\sum_k D(k,j)^2 - \frac{1}{2I^2}\sum_k\sum_l D(k,l)^2$$

(6.16)

La solution exacte quand elle existe

Supposons que les I échantillons existent, c'est-à-dire peuvent être représentés dans un espace, la matrice $\underline{S}$ des produits scalaires $\underline{y}_i.\underline{y}_j$ obtenus selon la relation (6.16) est symétrique (à cause de la commutativité

du produit scalaire). De plus, elle est égale au produit $\underline{Y}'.\underline{Y}$ de la matrice des coordonnées de ces échantillons $\underline{Y}$ par sa transposée $\underline{Y}'$, donc $\underline{S}$ est semi-définie positive.

Inversément, si $\underline{S}$ est symétrique semi-définie positive, la matrice $\underline{Y}$ des coordonnées des échantillons existent. En effet, la matrice symétrique semi-définie positive $\underline{S}$ peut s'écrire sous la forme :

$$\underline{S} = \underline{Q}'.\underline{A}^2.\underline{Q} \qquad (6.17)$$

où $\underline{A}^2$ est une matrice diagonale à valeurs propres positives et $\underline{Q}$ une matrice orthogonale.

Il suffit alors de prendre comme solution pour $\underline{Y}$:

$$\underline{Y}' = \underline{Q}'.\underline{A} \qquad (6.18)$$

En résumé, la condition nécessaire et suffisante pour que I échantillons donnés par leur distance mutuelle puissent être représentés dans un espace E^q ($q \leq I-1$) est que la matrice $\underline{S}$ formée selon (6.16) soit symétrique semi-définie positive.

La solution donnée par (6.18) donne une représentation des échantillons dans E^q. Notre objectif est de les placer au mieux dans un espace de dimension m petit. D'après le paragraphe précédent (§6.3), on peut déterminer les projections des échantillons représentés dans E^q sur le meilleur sous-espace F^m de dimension m donnée. Les échantillons i (i = 1, ...,I) sont alors représentés par les composantes des vecteurs :

$$\sqrt{\lambda_1}.\underline{v}_1 \quad , \quad \sqrt{\lambda_2}.\underline{v}_2 \quad , \quad \ldots \quad , \quad \sqrt{\lambda_m}.\underline{v}_m \qquad (6.19)$$

où λ_k et $\underline{v}_k$ sont les valeurs propres (par ordre décroissant) et vecteurs propres correspondants de $\underline{S}$.

En résumé, si la matrice $\underline{S}$ est symétrique et semi-définie positive, ses valeurs propres sont positives ou nulles. Dans l'espace E^q (q étant le nombre de valeurs propres de $\underline{S}$ non nulles) nous avons une représentation qui respecte exactement les distances entre les échantillons, mais q est généralement trop grand. Pour pouvoir effectuer une analyse visuelle sur les échantillons, il faut se contenter du sous-espace formé par les deux (éventuellement trois) premiers vecteurs propres associés aux deux plus

grandes valeurs propres. Ce sous-espace est le meilleur espace de dimension deux (ou trois) pour représenter les échantillons.

Dans le plan F^2, le i-ème échantillon est représenté par les i-èmes composantes des deux vecteurs suivants :

$$\sqrt{\lambda_1}.\underline{v}_1 \qquad \text{et} \qquad \sqrt{\lambda_2}.\underline{v}_2$$

La quantité d'information retenue par les deux premiers vecteurs propres, est mesurée par le facteur t_2 :

$$t_2 = \frac{\lambda_1 + \lambda_2}{\text{trace } (\underline{S})} \qquad (6.20)$$

Solution approchée

En général, la matrice $\underline{S}$ n'est pas semi-définie positive, même si les trois propriétés (6.14) sont vérifiées. Nous pouvons alors obtenir des valeurs propres négatives, dont les racines carrées donneraient des valeurs imaginaires. Par conséquent, il n'existe pas d'espace euclidien E^q de telle façon que les distances entre tous les échantillons soient exactement respectées.

Comme on s'intéresse seulement aux m (m égal à 2 ou 3) plus importants axes factoriaux, la question qui se pose est la suivante : quelle est la validité de la solution approchée qui consiste à choisir les m plus grandes valeurs propres positives et vecteurs propres associés de la matrice $\underline{S}$, dans le cas général où $\underline{S}$ n'est pas forcément semi-définie positive?

La réponse à cette question est donnée par ce qui suit. Cherchons une matrice $\underline{T}$ de rang m telle que

$$\underline{T} \approx \underline{S}$$

c'est-à-dire une matrice $\underline{T}$ telle que

$$|\underline{S} - \underline{T}|^2 = \sum_{i,j} (s(i,j) - t(i,j))^2 \quad \text{est minimum}$$

D'après la décomposition d'Eckart-Young /19/, la matrice $\underline{T}$ admet comme valeurs propres $\lambda_1, \lambda_2, \ldots, \lambda_m$ et $\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_m$ comme vecteurs propres;

Figure 6.1 : Comparaison des t_m calculés d'après (6.20) et (6.21)

a) Mot "zéro" par 25 locuteurs et 25 locutrices

Axe (m)	Valeur propre	t_m d'après (6.20)	t_m d'après (6.21)
1	291814.00	0.27	0.25
2	117362.05	0.38	0.36
3	86470.68	0.47	0.43
4	64921.58	0.53	0.49
5	54887.86	0.58	0.54
6	53712.87	0.63	0.58
7	43016.70	0.67	0.62
8	39846.72	0.71	0.66
9	32803.37	0.74	0.69
10	30764.01	0.77	0.71
.	.	.	.
.	.	.	.
26	-7748.93	0.99	0.93

b) Mot "deux" par 25 locuteurs et 25 locutrices

Axe	Valeur propre	t_m d'après (6.20)	t_m d'après (6.21)
1	420344.06	0.35	0.29
2	156584.16	0.48	0.40
3	143648.13	0.60	0.50
4	88835.56	0.67	0.57
5	69986.49	0.73	0.61
6	63463.53	0.78	0.66
7	54865.78	0.83	0.70
8	42050.13	0.86	0.73
9	33957.74	0.89	0.75
10	27314.53	0.91	0.77
.	.	.	.
.	.	.	.
14	-18398.72	0.96	0.83

c'est-à-dire, les mêmes que les m plus grandes valeurs propres positives et vecteurs propres associés de la matrice $\underline{S}$. Ainsi, $\underline{T}$ est une matrice symétrique semi-définie positive de rang m, qui approxime au mieux la matrice $\underline{S}$.

Pratiquement, il suffit de retenir comme solution les m vecteurs propres associés aux m plus grandes valeurs propres positives de $\underline{S}$. En fait, on se contentera de m = 2 (éventuellement m = 3).

Comme $\underline{S}$ peut avoir des valeurs propres négatives, nous définissons l'indice t_2 comme suit :

$$t_2 = \frac{\lambda_1 + \lambda_2}{\lambda_1 + \lambda_2 + \ldots + \lambda_m + \ldots + |\lambda_r| + \ldots + |\lambda_q|} \qquad (6.21)$$

Où $\lambda_r, \lambda_{r+1}, \ldots, \lambda_q$ sont des valeurs propres négatives. Précisons encore que s'il n'y a pas de valeurs propres négatives, (6.20) et (6.21) sont identiques. Les tests effectués sur des échantillons de parole montrent (voir exemples des figures 6.1a et 6.1b) qu'il y a peu de différence entre les t_2 calculés selon (6.20) et (6.21). Ceci est dû à l'effet négligeable des valeurs propres négatives de $\underline{S}$. Dans les tableau 6.1a et 6.1b, nous avons également indiqué la première valeur propre négative.

Autre approche

Une autre approche (méthode de Kruskal) consiste à chercher directement la meilleure configuration des I points dans l'espace de dimension fixée. La distance euclidienne entre deux vecteurs $\underline{y}_i = (y_i(1),\ldots,y_i(k),\ldots)$ et $\underline{y}_j = (y_j(1),\ldots,y_j(k),\ldots)$ dans F^m est

$$\sum_{k=1}^{m} (y_i(k) - y_j(k))^2$$

On cherche les $y_i(k)$ avec i = 1, ..., I et k = 1, ..., m qui minimisent la quantité suivante :

$$\sum_{\substack{i=1 \\ j=1}}^{I} \left(D(i,j)^2 - \sum_{k=1}^{m} (y_i(k) - y_j(k))^2 \right)^2$$

Il s'agit donc de minimiser une fonction à I x m inconnues. C'est un

problème de minimisation d'une fonction non linéaire sans contrainte . Il existe différents algorithmes pour trouver une solution approchée /57/. Toutefois, la solution obtenue par ces algorithmes risque de correspondre à un minimum local, alors que la solution proposée par la méthode factorielle correspond au minimum global.

6.5 Résumé des étapes pour arriver à une représentation dans le plan

1) Former la matrice $\underline{S}$ à l'aide des $D(i,j)$ connus selon la relation (6.16).

2) Déterminer les valeurs propres par ordre décroissant et les vecteurs propres correspondants. Comme la matrice $\underline{S}$ est symétrique, on peut utiliser la méthode de la puissance itérée valable pour les matrices symétriques /2/

$$\lambda_1 = \lim_{p \to \infty} \frac{| \underline{S}^p \underline{e} |}{| \underline{S}^{p-1} \underline{e} |} \quad , \ \underline{e} \text{ quelconque}$$

$$\underline{v}_1 = \lim_{p \to \infty} \frac{\underline{S}^p \underline{e}}{|\underline{S}^p \underline{e}|}$$

Pour trouver λ_2 et $\underline{v}_2$, on part de la matrice

$$\underline{S}^* = \underline{S} - \lambda_1 . \underline{v}_1 . \underline{v}'_1$$

et l'on applique de nouveau les deux relations précédentes, et ainsi de suite.

3) Si on veut une représentation dans le plan on forme les vecteurs :

$$\sqrt{\lambda_1} \cdot \begin{bmatrix} \cdot \\ \cdot \\ \underline{v}_1 \\ \cdot \\ \cdot \end{bmatrix} \qquad \sqrt{\lambda_2} \cdot \begin{bmatrix} \cdot \\ \cdot \\ \underline{v}_2 \\ \cdot \\ \cdot \end{bmatrix}$$

L'échantillon i est représenté par les i-èmes composantes des deux vecteurs ci-dessus.

4) Quantité d'information conservée sur les 2 premiers axes :

$$t_m = \frac{\lambda_1 + \lambda_2}{\lambda_1 + \lambda_2 + \ldots + \lambda_m + \ldots + |\lambda_r| + \ldots + |\lambda_q|} \qquad (6.21)$$

$(0 < t_m \leq 1)$

6.6 Analyse des échantillons multilocuteurs de parole

Dans ce qui suit diverses expériences sont effectuées sur des échantillons multilocuteurs de parole. Les échantillons sont des mots prononcés par des locuteurs ou des locutrices.

Les étapes suivantes sont effectuées :

- Pour un (ou plusieurs) mot(s) donné(s) prononcé(s) par divers locuteurs (trices), on obtient un ensemble de I échantillons. On détermine ensuite la matrice des distances D(i,j).

- Pour arriver à une représentation des échantillons dans un plan, on applique les étapes décrites dans le paragraghe 6.5

- On dessine les échantillons dans le meilleur plan F^2 que l'on a trouvé.

- Le pourcentage de la quantité d'information projetée est donné par les facteurs suivants : t_1 pour la meilleure droite de projection, t_2 pour le meilleur plan de projection, etc.

Rappelons encore que dans E^q défini par les vecteurs propres de S, toute l'information est conservée (il s'agit de l'information contenue dans la matrice des distances D(i,j)).

Les figures 6.2a, 6.2b et 6.2c montrent la projection des prononciations des mots "terminer", "zéro" et "deux" par 25 locuteurs et 25 locutrices sur le plan F^2 (le plan formé par le premier axe factoriel et le deuxième axe factoriel). Les locuteurs sont représentés par un triangle, et les locutrices par un cercle.
Les pourcentages d'information conservée valent :

fig.6.2a :	t_1 = 29%	t_2 = 39%
fig.6.2b :	t_1 = 25%	t_2 = 36%
fig.6.2c :	t_1 = 29%	t_2 = 40%

Les trois figures mentionnées ci-dessus montrent clairement l'existence d'un groupe masculin et d'un groupe féminin. Ainsi, nous pouvons déjà dire qu'il faut au moins deux groupes pour représenter un mot particulier.
La séparation entre les groupes masculins et féminins apparaît essentiellement sur le premier axe. Si l'on admet que la différence essentielle entre les hommes et les femmes se situe au niveau du fondamental (il y a évidemment d'autres différences), alors nous pouvons interpréter le premier axe factoriel comme étant "l'axe du fondamental". En fait, le

premier axe factoriel est une combinaison des principaux facteurs (le fondamental, les formants, etc) qui différencient un locuteur d'une locutrice.

Les figures 6.3a, 6.3b et 6.3c montrent la projection des prononciations des mots "cinq", "sept" et "huit" par 50 locuteurs sur le plan F^2.
Les pourcentages d'information conservée valent :

fig.6.3a : $t_1 = 17\%$ $t_2 = 27\%$
fig.6.3b : $t_1 = 24\%$ $t_2 = 38\%$
fig.6.3c : $t_1 = 18\%$ $t_2 = 31\%$

On peut observer la présence de groupes dans les trois figures. Ceci est dû naturellement à l'existence de "types" de prononciations au sein de la population constituée par les cinquantes locuteurs.
Dans le cas, par exemple, de la figure 6.3b (mot : "huit"), il y a des groupes qui représentent certainement le mot "hui" (le "t" n'a pas été prononcé ou a été perdu).

Les figures 6.4a, 6.4b et 6.4c montrent la projection sur le plan F^2 des couples de mots :

- "deux" et "huit"
- "cinq" et "sept"
- "six" et "sept"

Chaque fois les deux mots sont prononcés par 25 locuteurs (ou par 25 locutrices).
Les pourcentages d'information conservée valent :

fig.6.4a : $t_1 = 33\%$ $t_2 = 42\%$
fig.6.4b : $t_1 = 17\%$ $t_2 = 30\%$
fig.6.4c : $t_1 = 25\%$ $t_2 = 36\%$

Ces trois dernières figures montrent l'existence de classes de prononciations. Elles montrent aussi qu'une reconnaissance multilocuteur ne pourra se faire que si l'on arrive à regrouper les différentes locutions dans plusieurs classes caractéristiques des différentes fractions de la population.

En résumé, l'utilisation de l'analyse factorielle pour visualiser les prononciations de mots isolés par divers locuteurs a mis en évidence

l'existence de classes de prononciations pour un même mot.

Les classes les plus évidentes sont les 'classes masculines' et les "classes féminines"; elles sont probablement dues à des différences anatomiques. D'autre part, nous avons montré qu'il existe aussi des classes de prononciations au sein d'une population masculine.
Enfin nous avons montré (visuellement) que l'utilisation de plusieurs classes par mot pourrait diminuer la confusion entre les mots, donc améliorer la reconnaissance multilocuteur.
La détermination automatique des classes, ainsi que leur représentant, fait l'objet du chapitre suivant.

Fig.6.2 Projection des prononciations d'un mot par 25 locuteurs et 25 locutrices sur le plan déterminé par les deux premiers axes factoriels.

a) mot "terminer"
b) mot "zéro"
c) mot "deux"

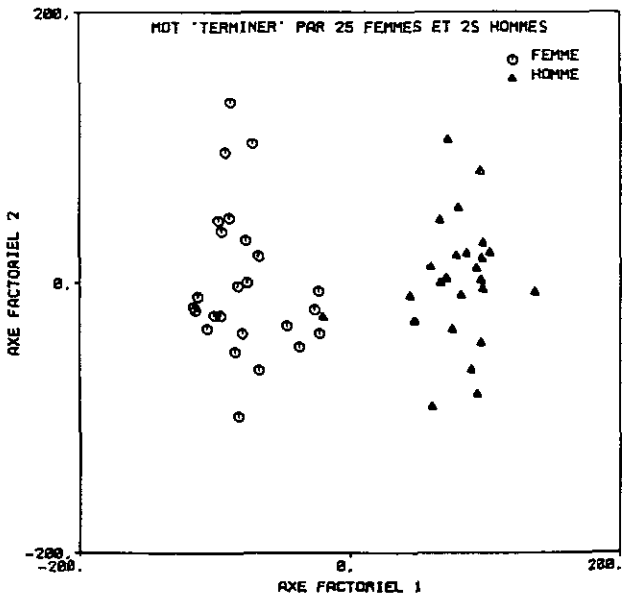


Fig.6.2a

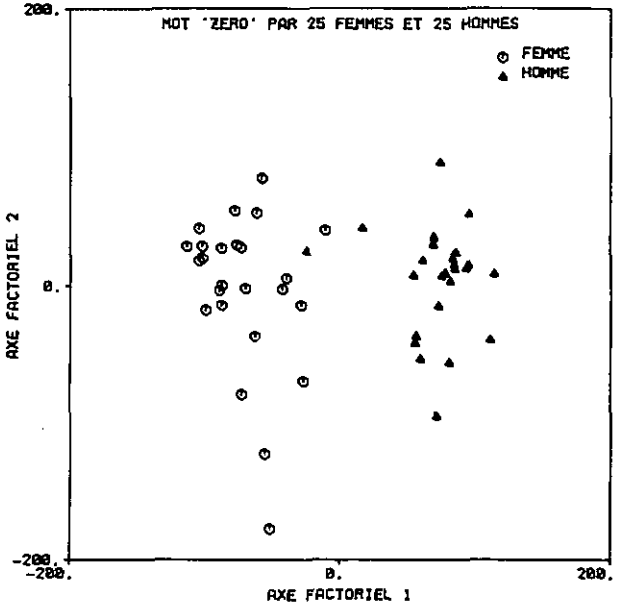


Fig.6.2b

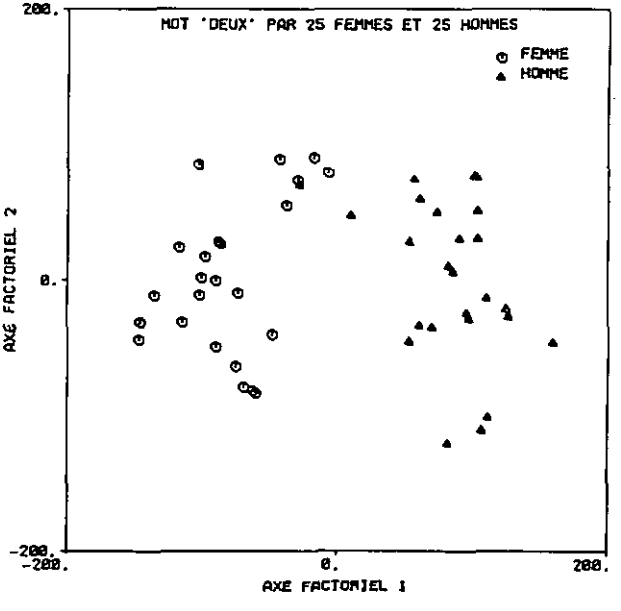


Fig.6.2c

Fig.6.3 Projection des prononciations d'un mot par 50 locuteurs masculins sur le plan déterminé par les deux premiers axes factoriels.

a) mot "cinq"
b) mot "sept"
c) mot "huit"

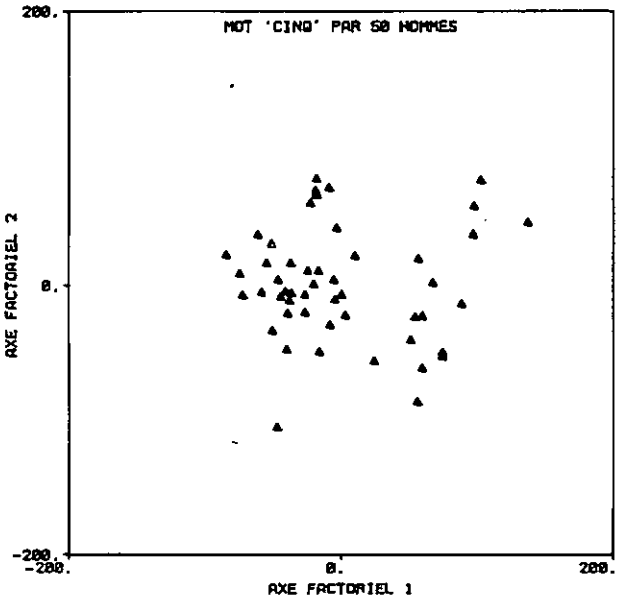


Fig.6.3a

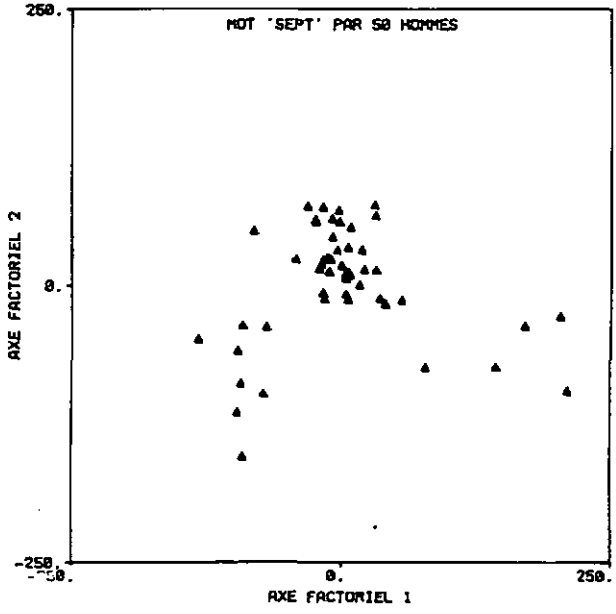


Fig.6.3b

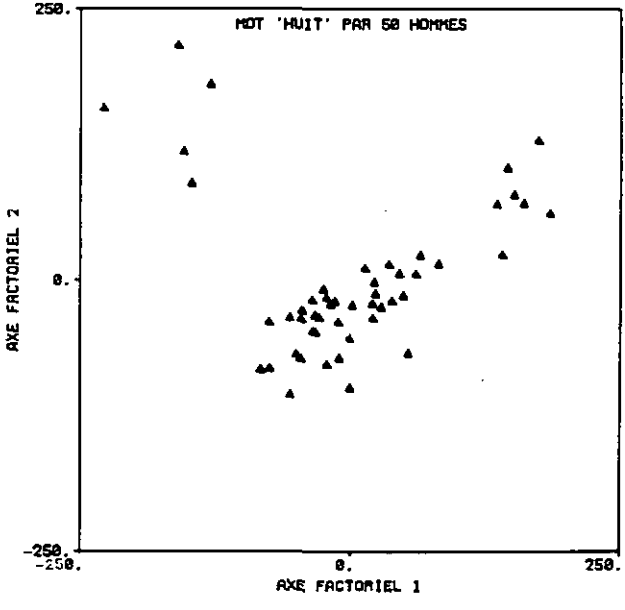


Fig.6.3c

Fig.6.4 Projection des prononciations de deux mots isolés par 25 locuteurs (ou 25 locutrices) sur le plan déterminé par les deux premiers axes factoriels.

a) mots "deux" et "huit" par 25 locuteurs
b) mots "cinq" et "sept" par 25 locutrices
c) mots "six" et "sept" par 25 locutrices

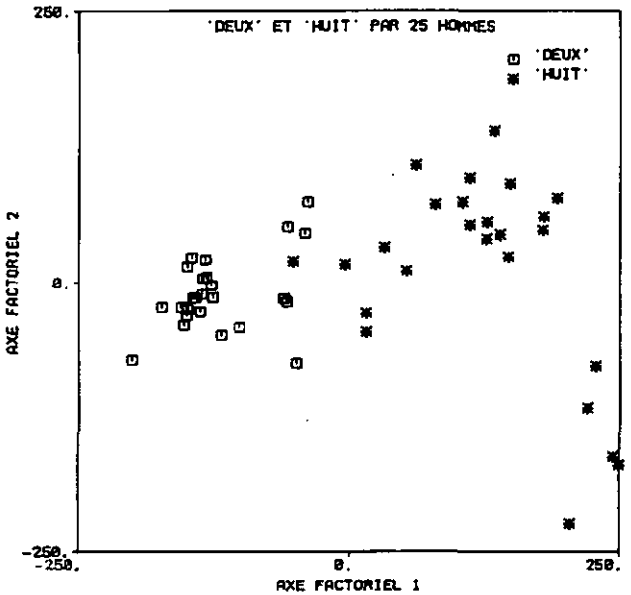


Fig.6.4a

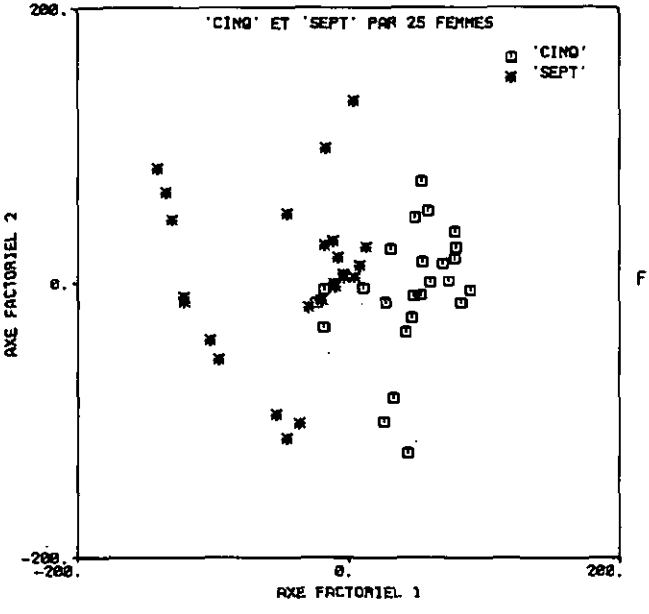


Fig.6.4b

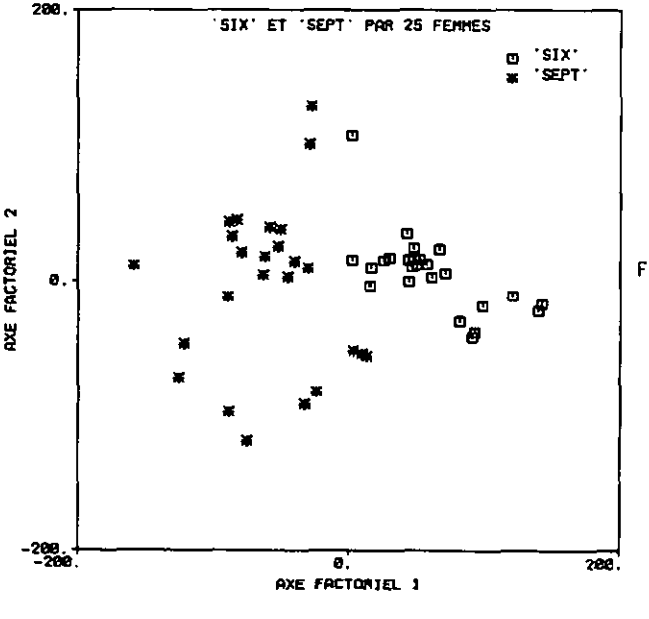


Fig.6.4c

7. Techniques de classification automatique appliquées à la reconnaissance multilocuteur

7.1 Introduction

Un système de reconnaissance multilocuteur de la parole vise à reconnaître une locution prononcée par n'importe quel locuteur. Le problème principal à résoudre est lié à la grande variation inter-locuteur.

D'après l'analyse que nous avons effectuée dans le chapitre précédent, il est raisonnable d'affirmer qu'au sein d'une population, il existe un certain nombre de classes de prononciations d'une locution donnée; chaque classe est caractéristique d'une fraction de la population. En conséquence, nous envisageons de décrire une locution avec un nombre limité de références, chacune étant le représentant d'une classe.

Pour identifier de manière automatique les classes mises en évidence par l'analyse factorielle, nous utilisons les techniques de classification automatique.

Il existe essentiellement deux techniques de classification :
1) dans les méthodes de partitionnement on cherche à regrouper les I éléments (dans notre cas, il s'agit des prononciations d'un mot par divers locuteurs) en K classes aussi homogènes que possible, et aussi différenciées les unes des autres que possible;
2) dans les méthodes hiérarchiques on cherche à regrouper les I éléments en une hiérarchie de classes de plus en plus grandes.

Les méthodes hiérarchiques sont intéressantes lorsqu'on cherche à identifier une certaine structure ou un certain ordre dans l'ensemble des éléments. Par contre, les méthodes de partitionnement s'imposent lorsqu'on cherche une caractérisation statistique des I éléments en un nombre limité de classes. C'est pourquoi seules les méthodes de partitionnement sont envisagées dans

ce travail.

Dans ce chapitre, nous développons et appliquons à la reconnaissance multilocuteur des mots isolés deux algorithmes de classification par partitionnement basés sur un critère d'optimisation. Une comparaison par rapport à d'autres algorithmes appliqués récemment à la reconnaissance multilocuteur des mots isolés /23/,/47/,/67/,/74/,/75/ est effectuée.

Nous commençons ce chapitre par la formulation générale du problème de la création de références (§7.2). Ensuite, nous expliquons le principe général des techniques de classification automatique par partitionnement (§7.3). Le paragraphe 7.4 introduit les notions mathématiques utilisées par la suite : définitions des métriques, définitions de fonctions d'homogénéité d'une classe, définitions du représentant d'une classe, etc.

Le paragraphe suivant (§7.5) est consacré à l'analyse de l'algorithme d'échange basé sur une fonction-critère (AEC). Différentes fonctions-critères sont définies, analysées et comparées. On compare aussi les différentes méthodes pour le choix du représentant d'une classe. Enfin, une comparaison avec l'algorithme 'regroupement autour de centre mobiles' (appelé aussi Basic Isodata ou k-means) /8/,/15/,/67/ est effectuée.

Le paragraphe 7.6 est consacré à l'analyse de l'algorithme séquentiel basé sur une fonction-critère (ASC). Une comparaison par rapport à l'algorithme séquentiel UWA /74/,/75/ est effectuée.

Les éléments parasites, dus à des mauvaises prononciations, peuvent perturber la classification. Une méthode pour leur élimination préalable est proposée dans §7.7.

Il est bien connu que, dans un vocabulaire donné, il y a des mots faciles et d'autres plus difficiles à reconnaître. Dans le paragraphe 7.8, on présente et teste une méthode permettant l'utilisation d'un nombre variable de références par mot, le but étant la réduction du nombre total de références pour un vocabulaire donné.

En conclusion (§7.9), nous faisons le point sur les différents algorithmes et méthodes traités dans ce chapitre.

7.2 Formulation de l'apprentissage multilocuteur

Si dans le cas d'un sytème monolocuteur une seule référence par mot est en général suffisante, dans le cas d'un système multilocuteur plusieurs

références par mot sont nécessaires.

Nous considérons le cas de la création de références multilocuteur pour chaque mot de manière séparée.

La création de références multilocuteur consiste à déterminer, à partir d'un ensemble de prononciations $\{X_i, i = 1,...,I\}$ d'un mot donné par divers locuteurs, un ensemble de références $\{R_k, k=1,...,K\}$, R_k étant le représentant d'une classe particulière C_k des prononciations du mot donné. Ainsi, pour un mot donné, on a :

$$\{X_i\} \longrightarrow \{C_k\} \longrightarrow \{R_k\} \text{ avec } i = 1,...,I \text{ et } k = 1,...,K$$

L'objectif recherché est de créer un ensemble de références valable pour le vocabulaire donné avec un nombre de représentants (ou références) par mot aussi faible que possible, et ceci pour une population la plus large possible. Le nombre K de références $\{R_k, k = 1...K\}$ par mot correspond aux classes $\{C_k, k = 1...K\}$ trouvées au sein de l'ensemble des prononciations $\{X_i, i = 1...I\}$.
Le nombre de locuteurs qui utiliseront le système de reconnaissance sera évidemment beaucoup plus grand que le nombre de références par mot.
A chaque classe correspond un représentant (qui est ensuite utilisé comme référence). Le représentant peut être soit un élément appartenant à la classe , soit un élément calculé selon certaines règles à partir des éléments de la classe.
Le nombre de références peut être le même pour chaque mot ou variable en fonction du mot (§7.8).

Exemples

Pour bien situer le problème, nous donnons dans le tableau qui suit quelques exemples de choix possibles. V prononciations du mot donné par chacun des L locuteurs sont utilisées pour créer K représentants par mot.

	Ensemble utilisé		Création de références	
	L	V	K (nombre de représentants par mot)	
mono.	1	1	1	R = X
	1	5	1	R = représentant des X_v
	1	5	5	$R_k = X_v$ (k,v = 1,...,5)
multi.	100	1	100	$R_k = X_i$ (k,i = 1,...,100)
	100	1	5	R_k, k=1,...,5, représentants déterminés par classification automatique

7.3 Principes des méthodes de classification par partitionnement

Les techniques de classification font appel à une démarche heuristique et non à un calcul formalisé. Alors que les valeurs des composantes des axes factoriels (chapitre 6) résultent de la solution d'équations, la détermination des classes se fait à partir d'une formulation heuristique.

Soient I le nombre d'éléments et K le nombre de classes. Le nombre de partitions différentes de I éléments en K classes est donné par le nombre de Stirling de deuxième espèce /40/ :

$$S(I,K) = \frac{1}{K} \sum_{j=1}^{K} (-1)^{K-j} \binom{K}{j} j^I$$

$$\approx K^{I-1} \qquad \text{pour I grand}$$

exemple :

$$I = 60 \ , \ K = 5 \longrightarrow S(60,5) = 1.7\ 10^{41}$$

Ce nombre est évidemment trop grand pour envisager d'utiliser une méthode, qui choisirait la meilleure partition après un dénombrement de tous les cas possibles. D'où la nécessité d'utiliser un algorithme heuristique et itératif.

Selon l'approche utilisée, nous pouvons diviser les algorithmes de classification par partitionnement en deux catégories :

1) les algorithmes parallèles (ou par échange) où les classes sont recherchées et produites simultanément;

2) les algorithmes séquentiels où les classes sont recherchées et produites l'une après l'autre.

Dans la première approche, où le nombre de classes est fixé à priori, l'algorithme parallèle cherche à améliorer la partition en transférant les éléments d'une classe à l'autre.

Dans la deuxième approche, où le nombre de classes n'est pas fixé à priori, l'algorithme séquentiel détermine à chaque itération la meilleure classe parmi un ensemble de classes-candidates; celles-ci sont déterminées à l'aide d'un seuil de distance. Les éléments de la classe ainsi déterminée sont ensuite retirés, et l'on recommence la même procédure jusqu'à ce qu'il n'y ait plus d'éléments à classer (remarquons que l'on peut aussi fixer le nombre de classes dans l'algorithme séquentiel : le seuil de distance pour lequel on obtient le nombre de classes désiré est alors déterminé itérativement).

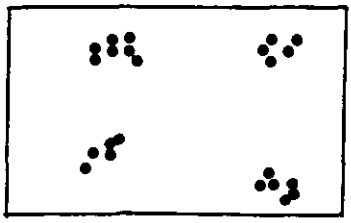


Fig.7.1a Ensemble des éléments d'un même mot pour différents locuteurs, dans l'espace des paramètres.

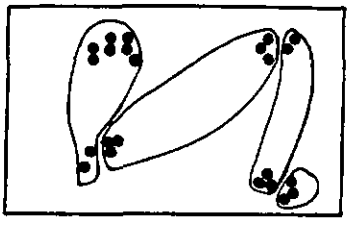


Fig.7.1b Initialisation de la partition initiale : choix aléatoire des K classes

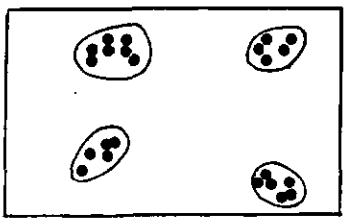


Fig.7.1c Résultat de l'algorithme de classification.

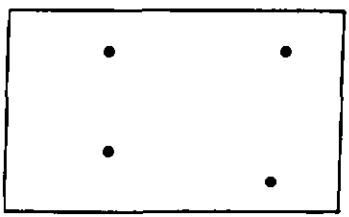


Fig.7.1d Choix des représentants de chaque classe.

Fig.7.1 Illustration des différentes étapes pour la création de la référence multilocuteur (algorithme parallèle).

7.4 Définitions

Soit C l'ensemble des éléments X_i, i = 1,...,I (c'est-à-dire l'ensemble des prononciations d'un même mot). On note C_k et C_l deux classes (deux sous-ensembles) de C. On supposera, dans tout ce qui suit, que les classes sont disjointes et non vides :

$$C_k = \emptyset \qquad \forall\, k = 1,\ldots,K$$
$$C_k \cap C_l = \emptyset \qquad \text{si } k \neq l$$
$$C = C_1 \cup C_2 \ldots \cup C_K$$

Soit $D(X_i,X_j)$ la distance entre les éléments X_i et X_j (distance dynamique entre les matrices correspondant aux éléments X_i et X_j), et soit n_k le nombre d'éléments dans la classe C_k.

7.4.1 Définition de la métrique $L_q(X_i,C_k)$

Soit C_k une classe :

$$C_k = \{\ldots,X_i,\ldots,X_j,\ldots\}$$

A l'élément X_i et la classe C_k nous associons la métrique générale suivante destinée à établir une mesure de distance entre l'élément X_i et le reste des éléments de la classe à laquelle il appartient :

$$L_q(X_i,C_k) = \left(\frac{1}{n_k-1} \sum_{X_j \in C_k} D(X_i,X_j)^q \right)^{1/q}$$

Cas particuliers importants :

1) q = 1 : $L_1(X_i,C_k)$ représente la moyenne des distances de X_i à tous les X_j.

2) q = 2 : $L_2(X_i,C_k)$ représente la distance quadratique moyenne (rms) de X_i à tous les X_j.

3) q = ∞ : $L_\infty(X_i,C_k)$ représente la distance de l'élément X_j^* à X_i, X_j^* étant l'élément le plus éloigné de X_i.

7.4.2 Définitions de fonctions d'homogénéité d'une classe

Nous postulons que, si l'homogénéité d'une classe est bonne, la fonction d'homogénéité est petite, et vice-versa. En d'autres termes, la fonction d'homogénéité sera d'autant plus petite que la dispersion des éléments de la classe est petite, et vice-versa.

Ci-dessous, sont définies les fonctions d'homogénéité $H(C_k)$ appelées :

- $Minlq(C_k)$
- $Moylq(C_k)$
- $Maxlq(C_k)$.

a) $Minlq(C_k) = \underset{X_i \in C_k}{Min} L_q(X_i, C_k)$

Cas particuliers :

1) $q = 1$: $Minl1(C_k)$ est le minimum des distances moyennes d'un élément à tous les autres au sein de la classe C_k.

2) $q = 2$: $Minl2(C_k)$ est le minimum des distances rms d'un élément à tous les autres au sein d'une classe C_k.

3) $q = \infty$: $Minl\infty(C_k)$ est le minimum des maximums des distances d'un élément à tous les autres au sein d'une classe C_k.

b) $Moylq(C_k) = \frac{1}{n_k} \sum_{X_i \in C_k} L_q(X_i, C_k)$

Notons que pour $q = 1$: $Moylq(C_k)$ est la distance moyenne entre tous les éléments de la classe C_k.

c) $Maxlq(C_k) = \underset{X_i \in C_k}{Max} L_q(X_i, C_k)$

Une fonction d'homogénéité H(Ck), telle qu'elle est définie ci-dessus, est une caractéristique d'une classe C_k. Si Dmin et Dmax sont respectivement le minimum et le maximum des distances entre les éléments d'une classe, on a toujours

$$Dmin \leq H(C_k) \leq Dmax$$

Les fonctions d'homogénéité définies sous a), b) et c) avec différentes valeurs de q vont nous servir à définir plus loin des fonctions-critères.

7.4.3 Définition du représentant d'une classe

Nous utilisons les définitions suivantes du représentant $G(C_k)$ d'une classe C_k :

a) Le représentant $Ga(C_k)$ est l'élément qui minimise la distance moyenne d'un élément à tous les autres :

$$Ga(C_k) = X_i^* \;/\; \min_{X_i \in C_k} L_1(X_i, C_k) = L_1(X_i^*, C_k)$$

b) Le représentant $Gb(C_k)$ est l'élément qui minimise la distance maximale d'un élément à tous les autres :

$$Gb(C_k) = X_i^* \;/\; \min_{X_i \in C_k} L_\infty(X_i, C_k) = L_\infty(X_i^*, C_k)$$

c) Le représentant $Gc(C_k)$ est la moyenne des éléments dans la classe C_k, c'est-à-dire l'élément dont la matrice est la matrice moyenne des matrices $\underline{X}_i$:

$$\underline{Gc}(C_k) = \frac{1}{n_k} \sum_i \underline{X}_i$$

7.5 Partitionnement par l'algorithme d'échange basé sur une fonction-critère (AEC)

7.5.1 Introduction

Soit $C_1, C_2, \ldots, C_K$ une partition de C, c'est-à-dire un ensemble de classes non vides, mutuellement disjointes et de réunion égale à C. Cette partition peut être bonne ou mauvaise. Pour le savoir, on définit une fonction $F(C_1,\ldots,C_K)$ qui mesure la qualité d'une partition (la fonction F est appelée par la suite fonction-critère). On supposera que la qualité de la partition augmente quand la fonction-critère diminue.

7.5.2 Description de l'algorithme AEC

L'algorithme d'échange basé sur une fonction-critère (AEC) produit de façon itérative et parallèle un nombre K de classes; K est fixé à priori.

On suppose donnée une partition de départ. Lors de chaque itération, un élément X_i est enlevé à sa classe C_k et transféré dans une autre classe C_l. On calcule ensuite la variation de la fonction-critère résultant du transfert de X_i de C_k dans C_l. En symbole :

$$\Delta F(X_i, C_k \rightarrow C_l) = F(C_1, \ldots, C_k - \{X_i\}, \ldots, C_l + \{X_i\}, \ldots) - F(C_1, \ldots, C_k, \ldots, C_l, \ldots)$$

Puisqu'il s'agit de minimiser la fonction-critère F, il convient de transférer X_i, si et seulement si :

$$\Delta F(X_i, C_k \rightarrow C_l) < 0$$

Pour avoir la plus grande décroissance possible de la fonction-critère, on affectera X_i à la classe $C_{l'}$ telle que :

$$\Delta F(X_i, C_k \rightarrow C_{l'}) = \min_l \Delta F(X_i, C_k \rightarrow C_l)$$

Comme les classes doivent être non vides, on ne peut pas enlever le dernier élément à une classe réduite à un seul élément. Le nombre de classes fournies est donc exactement égal à K. On s'arrête quand on ne peut plus échanger d'éléments.

Algorithme

1. Choisir une partition initiale $C_1, C_2, \ldots, C_K$
2. a) Trouver X_i appartenant à C pour lequel il existe une classe C_l telle que le transfert de X_i de sa classe C_k vers la classe C_l fasse diminuer la fonction-critère F :
   $\Delta F(X_i, C_k \rightarrow C_l) < 0$
   b) Arrêter si aucun élément n'a été trouvé :
   $\forall X_i \in C,\ l=1,\ldots,K : \Delta F(X_i, C_k \rightarrow C_l) \geq 0$
3. Transférer X_i dans la classe $C_{l'}$ qui donne la plus grande décroissance :

$$\Delta F(X_i, C_k \rightarrow C_{l'}) = \min_{l} \Delta F(X_i, C_k \rightarrow C_l)$$

4. Aller en 2.

7.5.3 Choix et étude des fonctions-critères

Rappelons que dans le cas de l'algorithme d'échange AEC, la fonction-critère est une grandeur qui mesure la qualité d'une partition, c'est-à-dire la division d'un ensemble de I éléments en K classes.

Pour trouver la meilleure partition, nous devons minimiser la fonction-critère, cette dernière étant à choisir de telle façon qu'elle mesure réellement la qualité de la partition. C'est pourquoi diverses fonctions-critères sont considérées et testées dans ce qui suit.

Nous définissons les trois catégories de fonctions-critères suivantes :

1) Fonctions-critères Fa

$$Fa_{1q} = \sum_k \text{Minlq}(C_k)$$

$$Fa_{2q} = \sum_k \text{Moylq}(C_k)$$

$$Fa_{3q} = \sum_k \text{Maxlq}(C_k)$$

2) Fonctions-critères Fb

$$Fb_{1q} = \sum_k \text{Minlq}(C_k).(n_k-1)$$

$$Fb_{2q} = \sum_k \text{Moylq}(C_k).(n_k-1)$$

$$Fb_{3q} = \sum_k \text{Maxlq}(C_k).(n_k-1)$$

3) Fonctions-critères Fc

$$Fc_{1q} = \sum_k \text{Minlq}(C_k).n_k.(n_k-1)$$

$$Fc_{2q} = \sum_{k} Moylq(C_k).n_k.(n_k-1)$$

$$Fc_{3q} = \sum_{k} Maxlq(C_k).n_k.(n_k-1)$$

Les fonctions-critères Fa sont directement égales à la somme des fonctions d'homogénéité $H(C_k)$.
Les fonctions-critères Fb sont égales à la somme des fonctions d'homogénéité $H(C_k)$ pondérées par n_k-1, n_k étant le nombre d'éléments de la classe C_k.
Enfin, les fonctions-critères Fc sont égales à la somme des fonctions d'homogénéité $H(C_k)$ pondérées par $n_k.(n_k-1)$.

Quelques cas particuliers :

- $Fa_{1\infty}$ est égal à la somme des minmax des classes C_k.

- lorsque le centre de gravité des éléments d'une classe C_k est défini par $Ga(C_k)$, alors la somme des inerties des classes est identique à Fb_{11} (l'inertie d'une classe étant définie par la somme des distances de chacun de ses éléments à leur centre de gravité).

- La somme des sommes de toutes les distances entre les éléments d'une classe est identique à Fc_{21}.

7.5.4 Analyse des différentes fonctions-critères

Pour étudier le comportement des diverses fonctions-critères utilisées dans l'algorithme d'échange AEC, nous avons effectué deux expériences EXP1 et EXP2; ce sont des tests de partitionnement où un partitionnement est considéré comme correct s'il réalise une classification des éléments par locuteur.

EXP1 : 5 locuteurs et 5 locutrices prononcent 6 fois chaque mot du vocabulaire choisi; nous disposons ainsi de 60 éléments (prononciations) pour chaque mot.
A partir d'une partition initiale de 10 classes choisie de façon aléatoire, nous déterminons pour chacune des fonctions-critères, la partition optimale à l'aide de l'algorithme AEC.

EXP2 : Cette expérience est, à une exception près, identique à EXP1. Alors que dans l'expérience précédente, chaque locuteur prononçait chaque mot le

même nombre de fois, dans cette expérience-ci, le nombre de prononciations n'est pas le même pour chaque locuteur. En effet, nous voulons savoir comment se comporte l'algorithme avec les diverses fonctions-critères dans le cas où les 10 locuteurs prononcent respectivement 8, 8, 6, 4, 4, 8, 8, 6, 4, 4 fois chacun des mots du vocabulaire (cas déséquilibré). Dans EXP2, on dispose également de 60 éléments par mot.

Soit Pr, la partition de référence, telle que chacune de ses classes ne contient que les éléments d'un seul locuteur. Si la fonction-critère est bien adaptée, et sous l'hypothèse que la variation intra-locuteur est plus faible que la variation inter-locuteur, l'algorithme doit trouver que la partition optimale est identique à Pr.

Influence de la partition initiale

L'algorithme d'échange est une méthode heuristique qui permet de trouver un optimum local, mais pas forcément l'optimum global. Il est intéressant de voir dans quelle mesure la partition optimale dépend de la partition initiale.

La figure 7.2 représente l'évolution de $Fc_{11}(Pf)/Fc_{11}(Pr)$ pour différentes partitions initiales. Pf est la partition (finale) trouvée par l'algorithme, Fc_{11} la fonction-critère définie précédemment.

Nous constatons que, pour différentes partitions initiales, l'algorithme donne des partitions ayant des valeurs de Fc_{11} proches ou identiques. Dans les résultats reportés sur la figure indiquée, la partition finale obtenue est identique à la partition Pr dans quatre cas sur cinq.

En conlusion, les tests effectués montrent que le résultat final est peu sensible au choix de la partition initiale.

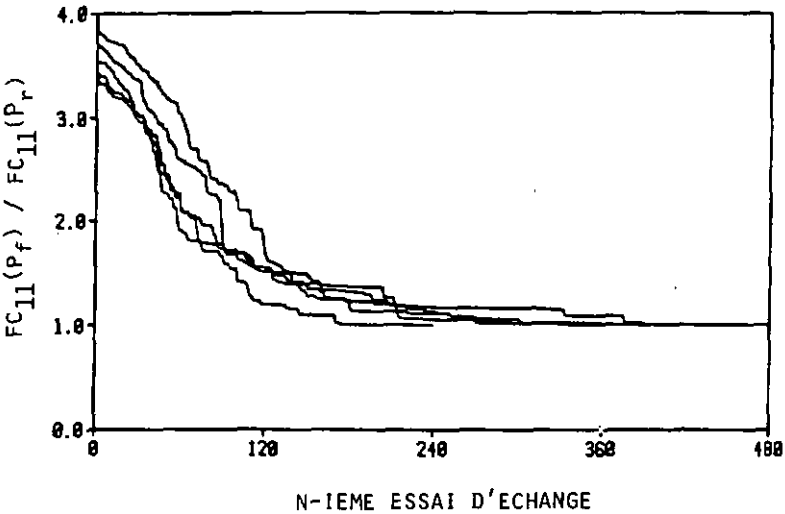


Fig.7.2 Effet des différents choix de la partition initiale sur la fonction-critère Fc_{11} (Mot : 'zéro' ; EXP1).

Taux de classification correcte pour les différentes fonctions-critères

Dans les tableaux de la figure fig.7.3 nous donnons, à titre d'exemple, les résultats détaillés du partionnement obtenu pour le mot 'zéro' dans le cas de l'expérience EXP1, pour les fonctions-critères Fa_{11} , Fb_{11} et Fc_{11}.

Nous pouvons faire les observations suivantes:

- Le partitionnement est effectué correctement pour les fonctions-critères Fb_{11} et Fc_{11}

- Pour la fonction-critère Fa_{11}, l'algorithme a tendance à mettre dans certaines classes très peu d'éléments, alors que le reste des éléments est mis dans une seule grosse classe. Souvent, les classes contiennent un seul élément. La partition obtenue pour de tels critères n'est pas utilisable.

Ainsi, le choix de la fonction-critère est très important. Dans ce qui suit nous allons analyser statistiquement le comportement des diverses fonctions-critères.

Mot : 'zéro'; EXP1

solution d'après AEC + Fa_{11}

classe \ locuteur	1	2	3	4	5	6	7	8	9	10
1	-	-	-	-	-	-	1	-	-	-
2	-	-	-	1	-	-	-	-	-	-
3	-	-	-	-	-	1	-	-	-	-
4	-	-	1	-	-	-	-	-	-	-
5	-	-	-	5	3	-	-	-	-	-
6	-	4	-	-	-	-	-	-	-	-
7	3	-	-	-	-	-	-	-	-	-
8	-	-	-	-	1	-	-	-	-	-
9	3	2	5	-	2	5	5	5	6	6
10	-	-	-	-	-	-	-	1	-	-

solution d'après AEC + Fb_{11}

classe \ locuteur	1	2	3	4	5	6	7	8	9	10
1	-	-	-	-	-	-	6	-	-	-
2	-	6	-	-	-	-	-	-	-	-
3	-	-	-	-	-	6	-	-	-	-
4	-	-	6	-	-	-	-	-	-	-
5	-	-	-	6	2	-	-	-	-	-
6	-	-	-	-	-	-	-	-	-	6
7	6	-	-	-	-	-	-	-	-	-
8	-	-	-	-	4	-	-	-	-	-
9	-	-	-	-	-	-	-	-	6	-
10	-	-	-	-	-	-	-	6	-	-

solution d'après AEC + Fc_{11}

classe \ locuteur	1	2	3	4	5	6	7	8	9	10
1	-	-	-	-	-	-	6	-	-	-
2	-	-	-	6	-	-	-	-	-	-
3	-	-	-	-	-	6	-	-	-	-
4	-	6	-	-	-	-	-	-	-	-
5	-	-	-	-	-	-	-	6	-	-
6	-	-	-	-	-	-	-	-	6	-
7	6	-	-	-	-	-	-	-	-	-
8	-	-	-	-	6	-	-	-	-	-
9	-	-	-	-	-	-	-	-	-	6
10	-	-	6	-	-	-	-	-	-	-

Fig.7.3 Exemples de partitionnement

Dans les expériences EXP1 et EXP2, l'ensemble des éléments pour chaque mot est constitué par les prononciations de L locuteurs, chaque locuteur ayant prononcé V_l ($l = 1,\ldots,L$) fois le mot. On cherche L classes à l'aide de l'algorithme AEC en partant d'une partition initiale quelconque.
Comme nous l'avons déjà indiqué plus haut : si la variation intra-locuteur est plus faible que la variation inter-locuteur (ceci n'est pas toujours vrai), la partition idéale est telle que dans chacune des classes, il n'y a que des éléments appartenant à un seul locuteur.
Soit donc la partition Pr de L classes telle que dans chacune de ses classes il n'y a des éléments que d'un locuteur. Pr n'est pas forcément optimale au sens du critère choisi.

Nous allons définir un facteur TCC qui mesure l'écart d'une partition par rapport à Pr. Il est clair que les partitions proches de Pr sont plus acceptables que celles qui s'en écartent fortement. Le facteur TCC permet de 'juger' la fonction-critère choisie.

Soit $Vmax(C_k)$ le nombre maximum d'éléments appartenant à un même locuteur au sein d'une classe C_k. On peut dire que la classe C_k est associée à ce dernier locuteur. Le nombre d'éléments mis incorrectement dans la classe C_k est :

$$n_k - Vmax(C_k)$$

Le nombre total d'erreurs est

$$\sum_k (n_k - Vmax(C_k))$$

c'est-à-dire :

$$I - \sum_k Vmax(C_k)$$

Le taux de classification correcte est alors défini comme suit :

$$TCC = (\sum_k Vmax(C_k)) / I$$

TCC est toujours inférieur ou égal à 1 (ou 100%). Il est égal à 1 lorsque la partition trouvée est indentique à la partition Pr.

La figure 7.4 donne la moyenne des taux de classification correcte sur tout le vocabulaire pour les diverses fonctions-critères (EXP1 et EXP2).

Les résultats indiqués dans les figures 7.4 montrent que :

- Les fonctions-critères Fa_{pq} - c'est-à-dire celles basées sur la somme des fonctions d'homogénéité - donnent de 'mauvaises' partitions : les partitions obtenues, bien qu'optimales selon le critère choisi, ne sont pas utilisables. Comme nous l'avons déjà illustré dans la figure 7.3, l'algorithme a tendance à mettre très peu d'éléments dans la plupart des classes, et le reste dans une grosse classe.

- Les fonctions-critères Fb_{pq} - c'est-à-dire celles basées sur la somme des fonctions d'homogénéité pondérées par le nombre d'élément de la classe - donnent, en général, des partitions proches (ou identiques) de la partition Pr. Il existe ainsi des fonctions-critères du type Fb qui sont bien adaptées dans le cas d'une classification sur les échantillons de la parole.

- Les fonctions-critères Fc_{pq} - c'est-à-dire celles basées sur la somme des fonctions d'homogénéité pondérées par $n_k.(n_k-1)$ - donnent en général aussi des partitions proches (ou identiques) de la partition Pr. Il existe donc des fonctions du type Fc qui sont bien adaptées dans le cas d'une classification sur les échantillons de la parole.

De façon générale, nous pouvons tirer la conclusion suivante : lors de la définition de la fonction-critère d'une partition, il convient de pondérer la fonction d'homogénéité d'une classe par le nombre n_k de ses éléments (ou par $n_k.(n_k-1)$).
Notons que, si dans nos définitions la pondération apparait de façon explicite, il est possible de définir une fonction-critère où la pondération apparait de façon implicite; par exemple en définissant la fonction-critère égale à la somme des sommes des distances des éléments d'une classe à leur centre de gravité (si le centre de gravité est définie par $Ga(C_k)$ cette fonction-critère est la même que Fb_{11}).

a) EXP1

Fa_{pq}

p \ q	1	2	∞
1	0.30	0.30	0.41
2	0.32	0.32	0.30
3	0.36	0.30	0.34

b) EXP2

Fa_{pq}

p \ q	1	2	∞
1	0.30	0.39	0.40
2	0.32	0.31	0.27
3	0.35	0.40	0.33

Fb_{pq}

p \ q	1	2	∞
1	0.84	0.80	0.73
2	0.82	0.75	0.75
3	0.57	0.57	0.66

Fb_{pq}

p \ q	1	2	∞
1	0.83	0.59	0.78
2	0.82	0.71	0.75
3	0.71	0.60	0.70

Fc_{pq}

p \ q	1	2	∞
1	0.86	0.84	0.51
2	0.86	0.82	0.55
3	0.41	0.55	0.60

Fc_{pq}

p \ q	1	2	∞
1	0.80	0.78	0.52
2	0.88	0.80	0.51
3	0.52	0.55	0.41

Fig.7.4 : Taux de classification correcte (TCC) pour les différentes fonctions-critères. Les taux indiqués sont des taux moyens sur tout le vocabulaire; a) EXP1 ; b) EXP2.

7.5.5 Tests de reconnaissance multilocuteur

A partir d'un ensemble de I éléments (I prononciations d'un mot par divers locuteurs), on crée une partition de K classes pour chaque mot. Puis, on détermine pour chaque classe un représentant. Les K représentants forment la référence multiple du mot considéré.

TST D : Le vocabulaire de test choisi est composé des mots suivants (nous associons à chaque mot un numéro) :

mots	numéro
en-avant	1
en-arrière	2
terminer	3
zéro	4
un	5
deux	6
trois	7
quatre	8
cinq	9
six	10
sept	11
huit	12
neuf	13

L'ensemble des éléments utilisés pour la création de la référence multilocuteur pour un mot donné est constitué des prononciations de ce mot par 25 hommes et 25 femmes. Un autre ensemble, constitué par 3 prononciations de chacun des mots du vocabulaire par 5 hommes et 5 femmes, est utilisé pour les tests de reconnaissance. Notons bien que le locuteur-référence est toujours différent du locuteur-test.

Les valeurs des paramètres (Annexe A) sont les suivantes :

NAMP = Z3
NTMP = 1 N = 20
QUANT = 1 Nb = 1 r0 = 0 r1 = 2
DMAT = 3 DVEC = 2 RANGE = 4 CDF = 0

En cas de modification de la valeur d'un paramètre, nous l'indiquerons explicitement.

La figure 7.5 donne les taux d'erreur en fonction du nombre de références par mot (quantification avec un nombre de bits égal à un).
Il y a d'une part les résultats obtenus avec des références choisies de façon aléatoire, et d'autre part, les résultats obtenus avec des références sélectionnées par l'algorithme AEC avec les fonctions-critères Fb_{11} et Fc_{21}.
Le représentant d'une classe C_k est $Ga(C_k)$.

Nous pouvons tirer les conclusions suivantes :

- par rapport à un choix aléatoire, la supériorité des performances obtenues avec des références sélectionnées par l'algorithme de classification est évidente.
- nous constatons la décroissance rapide du taux d'erreur en fonction du nombre de références par mot (ou nombre de classes par mot).
- bien que le nombre de classes soit assez petit (inférieur à 6) par rapport au nombre de locuteurs (50) participant à l'apprentissage, on obtient de bonnes performances de reconnaissance multilocuteur.

La figure 7.6 compare les résultats de reconnaissance obtenus avec un nombre de bits égal à un (Nb = 1) et un nombre de bits égal à quatre (Nb = 4).
Lorsque le nombre de références par mot est compris entre 2 et 5, on observe une supériorité des performances obtenues avec Nb = 4 par rapport aux performances obtenues avec Nb = 1.
Mais, pour une taille de mémoire donnée, il vaut mieux prendre un plus grand nombre de références avec Nb = 1, plutôt qu'un petit nombre de références avec Nb = 4. En effet, nous pouvons constater (fig.7.6) que le taux d'erreur est deux fois plus petit lorsque on a 4 références par mot avec Nb = 1, par rapport à une référence par mot avec Nb = 4.
Tous les tests qui suivront seront effectués avec Nb = 1. (Rappelons que l'un des buts poursuivis dans ce travail est de minimiser la quantité de données nécessaire pour représenter un mot).

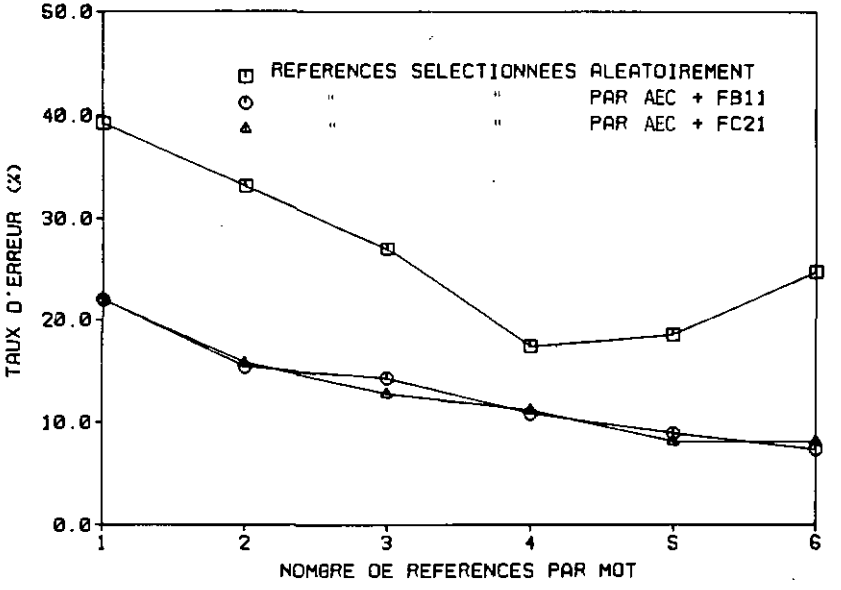


Fig.7.5 Taux d'erreur en reconnaissance multilocuteur en fonction du nombre de classes (algorithme de classification AEC, représentant d'une classe Ga(Ck)).

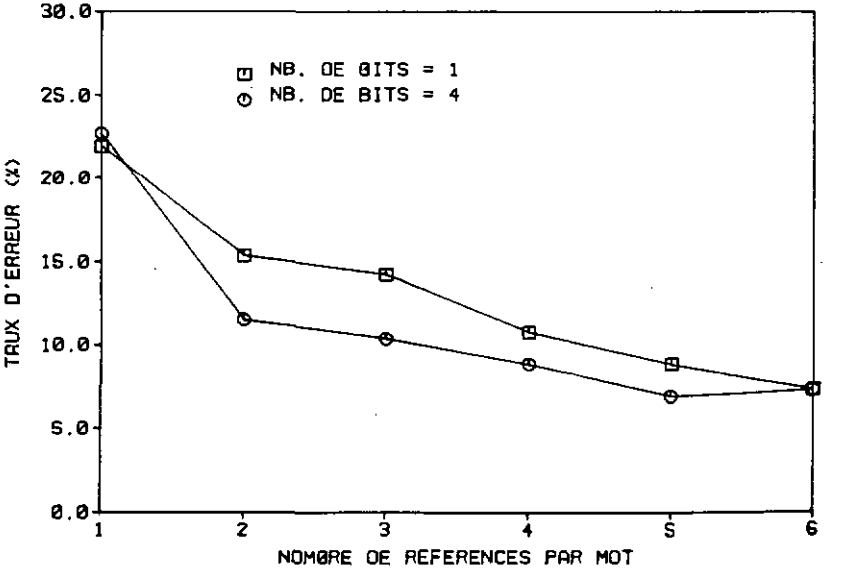


Fig.7.6 Taux d'erreur en reconnaissance multilocuteur en fonction du nombre de classes; comparaison entre Nb = 1 et Nb = 4; (algorithme de classification AEC avec Fb_{11}, représentant d'une classe $Ga(C_k)$).

7.5.6 Comparaison des choix du représentant d'une classe

Soit C_k une classe et $R(C_k)$ son représentant. Les trois méthodes suivantes sont comparées (§7.4.3) :

1) $R(C_k) = Ga(C_k)$: le représentant d'une classe est l'élément qui minimise la distance moyenne d'un élément à tous les autres.

2) $R(C_k) = Gb(C_k)$: le représentant d'une classe est l'élément qui minimise la distance maximale d'un élément à tous les autres (c'est le minmax bien connu).

3) $R(C_k) = Gc(C_k)$: Le représentant est l'élément égal à la moyenne des éléments dans la classe C_k, c'est-à-dire la matrice moyenne des matrices $\underline{X}_i$.

La figure 7.7 donne le taux d'erreur de reconnaissance multilocuteur en fonction du nombre de classes dans les trois cas. Les résultats montrent la supériorité de $Gc(C_k)$ par rapport à $Ga(C_k)$ et $Gb(C_k)$. Ces deux dernières méthodes donnent des résultats comparables avec toutefois, un léger avantage pour $Ga(C_k)$.
Dans un cas pratique, la méthode $Gc(C_k)$ est lourde à appliquer et nécessite la création d'un nouvel élément. Les deux premières ($Ga(C_k)$ et $Gb(C_k)$) sont beaucoup plus simples à appliquer, et surtout, utilisent un élément qui existe déjà.

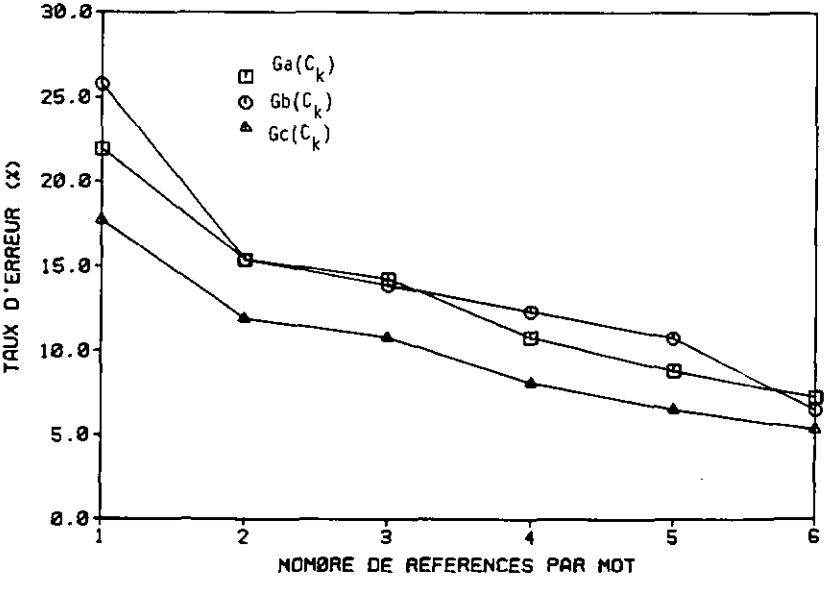


Fig.7.7 Comparaison des différentes méthodes pour le choix du représentant d'une classe (algorithme de classification AEC avec Fb_{11}).

7.S.7 Comparaison entre l'algorithme AEC et l'algorithme BI

Dans l'algorithme "Basic Isodata" (BI) /17/, appelé aussi k-means, (en français 'regroupement autour de centres mobiles'), tous les éléments sont transférés, à chaque itération, dans la classe dont le centre de gravité est le plus proche (le centre de gravité d'une classe est le centre de gravité de ses éléments). Quand tous les éléments ont été affectés, on calcule les nouveaux centres de gravité et l'on passe à l'itération suivante. On s'arrête quand on ne peut plus réaffecter d'éléments.

Algorithme BI

1. Choisir une partition initiale $C_1, C_2, \ldots, C_K$
2. Calculer les centres de gravité
3. Transférer tout X_i dans la classe dont le centre de gravité est le plus proche
4. Si aucun X_i n'a été transféré : Arrêt
   Sinon : Aller en 2.

Le lien entre l'algorithme AEC et l'algorithme BI

Dans l'algorithme d'échange AEC décrit au paragraphe 7.S.2, la variation de la fonction-critère qui résulte du transfert d'un élément X_i appartenant à C_k vers C_l est :

$$\Delta F(X_i, C_k \rightarrow C_l)$$

Supposons que la fonction-critère choisie est Fb_{11}. Celle-ci mesure la somme des sommes des distances des éléments d'une classe à son centre de gravité lorsque ce dernier est défini par $Ga(C_k)$ (ici le centre de gravité d'une classe a la même signification que le représentant d'une classe); Fb_{11} peut être réécrite sous la forme :

$$Fb_{11} = \sum_k \sum_{X_i \in C_k} D(X_i, Ga(C_k))$$

Si le transfert d'un élément X_i de C_k vers C_l ne modifie que de façon insignifiante les centres de gravité des classes C_k et C_l, alors la variation de la fonction-critère est :

$$\Delta F(X_i, C_k \rightarrow C_l) = D(X_i, Ga(C_l)) - D(X_i, Ga(C_k))$$

Dans l'algorithme AEC, on affecte ensuite X_i à la classe C_l, pour laquelle $D(X_i, G(C_l))$ est minimum. En d'autres termes, on affecte X_i à la classe dont le centre de gravité est le plus proche de X_i.

Dans une grande mesure, l'algorithme BI peut être vu comme une forme particulière de l'algorithme AEC (lorsqu'il utilise par exemple la fonction-critère Fb_{11}) où la mise à jour des centres de gravité se ferait à un rythme plus lent. Plus précisément, la mise à jour se ferait seulement après que chaque élément ait été exactement une fois candidat à l'échange). Et inversement, l'algorithme AEC peut être vu comme une forme particulière de l'algorithme BI, où la mise à jour des centres de gravité se ferait après chaque transfert.

Le tableau suivant donne les taux de classification correcte TCC obtenus dans les expériences EXP1 et EXP2 par :

- l'algorithme AEC avec Fb_{11} et $Fb_{1\infty}$
- l'algorithme BI avec le centre de gravité défini par $Ga(C_k)$ et $Gb(C_k)$

TCC

		EXP1	EXP2
AEC	critère Fb_{11}	0.84	0.83
	critère $Fb_{1\infty}$	0.73	0.78
BI	c. gravité Ga(Ck)	0.67	0.74
	c. gravité Gb(Ck)	0.65	0.70

Ce tableau montre clairement la supériorité des résultats de classification obtenus par l'algorithme AEC par rapport à l'algorithme BI.

La figure 7.8 donne les résultats de reconnaissance multilocuteur dans les

cas où les références sont sélectionnées par :

a) AEC + Fb_{1I} avec Ga(Ck)

b) AEC + $Fb_{1\infty}$ avec Gb(Ck)

c) BI avec Ga(Ck)

d) BI avec Gb(Ck)

Nous pouvons remarquer que, lorsque le nombre de références par mot est supérieur à 2, les performances obtenues par l'algorithme AEC sont meilleurs que celles obtenues par l'algorithme BI.

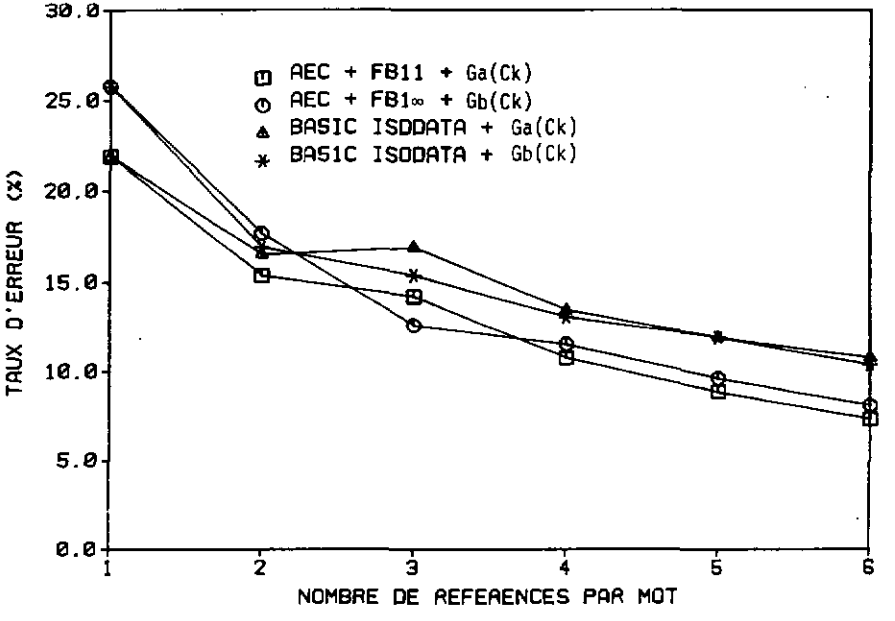


fig.7.8 Comparaison des résultats de reconnaissance multilocuteur lorsque les références sont sélectionnées d'une part par l'algorithme AEC et d'autre part par l'algorithme BI.

7.5.8 Le point sur l'algorithme AEC

Nous pouvons tirer les conclusions suivantes :

- L'algorithme de classification par échange basé sur l'optimisation d'une fonction-critère judicieusement choisie permet de trouver les groupes de prononciations au sein d'une population. Les fonctions-critères du type Fb_{pq} ou Fc_{pq} (avec p = 1, 2 et q = 1, 2) conviennent.

- Les performances de reconnaissance obtenues avec des références sélectionnées par l'algorithme de classification sont nettement meilleures que celles obtenues avec comme référence un choix aléatoire d'éléments.

- Avec un nombre de références égal à 5 (ces dernières étant sélectionées par l'algorithme AEC avec Fb et $Ga(C_k)$), on ramène le taux d'erreur en reconnaissance multilocuteur à environ 8%.

- Les résultats obtenus avec l'algorithme AEC sont meilleurs à ceux obtenus avec l'algorithme BI.

7.6 Partitionnement par l'algorithme séquentiel basé sur une fonction-critère (ASC)

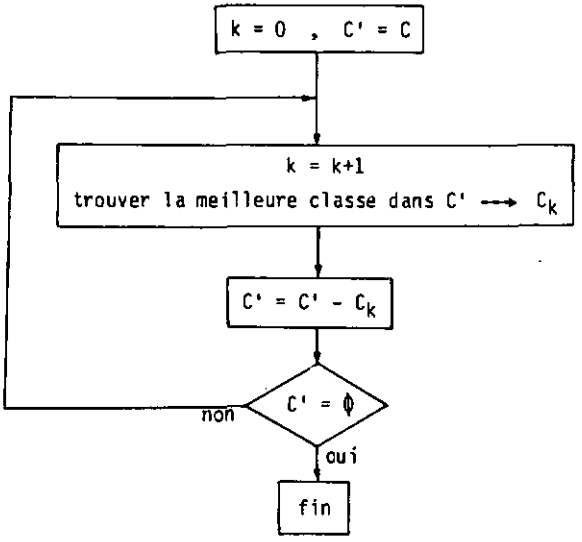
7.6.1 Principe

Dans l'algorithme ASC on procède de façon itérative et séquentielle : les classes sont créées l'une après l'autre; à chaque itération on cherche la meilleure classe parmi les classes-candidates. Les éléments de la meilleure classe sont alors soustraits, et l'on continue la procédure jusqu'à ce qu'il n'y ait plus d'éléments à classer.

A chaque itération intervient un seuil de distance, pour la création des classes-candidates.

Dans l'algorithme séquentiel, le nombre de classes n'est en principe pas fixé à priori; par contre, le seuil de distance doit l'être. On peut toutefois fixer le nombre de classes à priori, mais il faut alors déterminer itérativement le seuil de distance afin d'obtenir le nombre de classes désiré.

Le principe de l'algorithme séquentiel est illustré par l'organigramme suivant :



7.6.2 Description de l'algorithme ASC

A chaque itération, des classes-candidates sont créées. Une classe-candidate est définie par l'ensemble des éléments X_j appartenant à C' (C' est l'ensemble des éléments qui restent à classer) situés autour d'un élément X_i, et telle que la distance $D(X_i,X_j)$ est inférieure à un seuil de distance T. Il y a autant de classes-candidates que d'éléments X_i dans C'.

La meilleure classe est définie comme suit : parmi l'ensemble des classes-candidates, on cherche celle qui optimise une fonction-critère.

L'algorithme ASC

Soit $A(X_i)$ la classe-candidate associée à X_i pour un seuil de distance donné T :

$$A(X_i) = \{ X_j \in C' \ / \ D(X_i,X_j) < T \}$$

et soit $F(A(X_i))$ une fonction-critère (remarquons que la fonction-critère est définie sur une seule classe). On supposera que F est petit si l'homogénéité est grande et vice-versa. La description complète de l'algorithme ASC est la suivante:

1. Initialisation :

   $k = 1$ (k-ième classe)
   $C' = C$ (le reste est égal à l'ensemble de tous les échantillons)

2. Pour tout $X_i \in C'$ trouver la classe-candidate $A(X_i)$:

   $$A(X_i) = \{ X_j \in C' \text{ tel que } D(X_i,X_j) < T \}$$

3. Déterminer la classe-candidate qui minimise la fonction-critère F :

   $$C_k = A(X_i^*) \ / \ F(A(X_i^*)) \leq F(A(X_i)) \quad \forall \ X_i \in C'$$

4. $C' = C - C_k$

5. Si $C' \neq \phi$: $k = k+1$ et aller en 2.
   Sinon : fin

7.6.3 Choix et étude de fonctions-critères pour l'algorithme ASC

Choix

Les fonctions-critères sont définies sur la base des fonctions d'homogénéité Minl1, Minl∞ , Moyl1 (§7.4.2) et le nombre d'élements n_A de la classe-candidate A.

F1 = Minl1 (A)
F2 = Minl∞ (A)
F3 = Moyl1 (A)
F4 = Minl1 (A) + Minl∞ (A)
F5 = (Minl1 (A) + Minl∞ (A)) / n_A
F6 = 1 / n_A

Le choix de ces fonctions est motivé comme suit :

- Avec F1, F2, F3, F4 on peut comparer différentes fonctions-critères qui sont directement basées sur des fonctions d'homogénéité.

- Avec F4, F5 et F6 on peut évaluer les effets séparés ou combinés d'une fonction d'homogénéité définie par Minl1 (A) + Minl∞ (A) et du nombre d'éléments n_A de la classe.

Nous cherchons des classes qui, en plus du fait qu'elles doivent être très homogènes, contiennent le plus grand nombre possible d'éléments. C'est pourquoi nous avons défini la fonction F5 qui est proportionnelle à Minl1 (A) + Minl∞ (A) et inversément proportionnelle au nombre d'éléments n_A de la classe-candidate A.
Enfin, pour voir l'effet isolé de n_A, nous avons défini F6 qui est égale à l'inverse du nombre d'éléments n_A

Etude des fonctions-critères

Soit l'expérience suivante :

EXP3 : 5 locuteurs et 5 locutrices prononcent 6 fois chaque mot du vocabulaire choisi (idem à EXP1); on dispose ainsi de 60 échantillons pour chaque mot. Pour un seuil de distance donné T, on cherche alors une partition à l'aide de l'algorithme ASC ayant une fonction-critère comme

définie ci-dessus.

Comme pour l'algorithme AEC, on prend comme partition de référence la partition Pr où toutes les classes ne contiennent que les éléments d'un seul locuteur. Pour un seuil de distance T judicieusement choisi et sous l'hypothèse que la variation intra-locuteur est plus faible que la variation inter-locuteur, on doit (si l'algorithme est efficace) obtenir la partition Pr, c'est-à-dire 10 classes contenant chacune 6 échantillons appartenant à un même locuteur.
Les autres partitions seront jugées par rapport à Pr au moyen du taux de classification correcte TCC défini ci-après.

Définition du taux de classification correcte TCC

Dans le cas de EXP3, pour un seuil donné T, le nombre de classes trouvé n'est pas forcément égal au nombre de locuteurs. Les classes obtenues sont ordonnées de telle façon que l'on ait :

$$Card(C_1) \geq Card(C_2) \geq \ldots \geq Card(C_K)$$

où $Card(C_k)$ est le cardinal de C_k (nombre d'éléments dans la classe), et K le nombre de classes trouvé avec le seuil donné T.
Soit L le nombre de locuteurs ayant participé au test EXP3 et I le nombre total d'éléments. TCC est alors calculé comme suit (voir §7.5.4) :

$$TCC = \sum_{k=1}^{K'} Vmax(C_k) \; / \; I$$

$$K' = Min \; (L,K)$$

Ainsi, dans la définition de TCC, on ne tient compte que des K' plus grandes classes.

Résultats

La figure 7.9 donne le taux TCC en fonction du seuil de distance T pour les trois fonctions F4, F5 et F6 (mot choisi : 'zéro', EXP3). Les conclusions suivantes s'imposent :
- supériorité de F4 par rapport à F5 et F6
- pratiquement pas de différence entre F5 et F6; ceci est dû à l'effet prépondérant de $1/n_A$ dans F5.

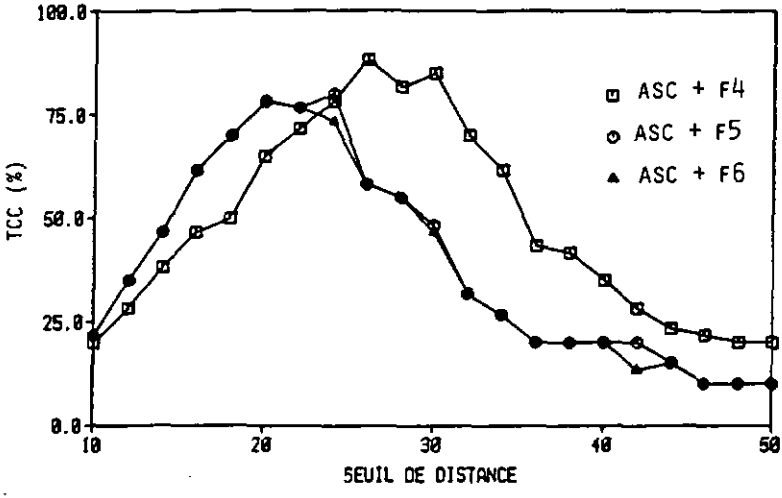


Fig.7.9 : Taux TCC dans le cas de l'algorithme ASC avec F4, F5, F6 (mot : zéro ; EXP3)

Le tableau de la figure 7.10 donne, pour chacune des fonctions-critères F1, F2, F3, F4, F5 et F6, la moyenne des TCC maximum (TCCmax) des différents mots du vocabulaire, ainsi que l'écart-type de TCCmax (le TCCmax est obtenu en faisant varier le seuil par petit pas).

Nous pouvons constater :

- que les fonctions-critères F1, F2, F3 et F4, basées uniquement sur des fonctions d'homogénéité, donnent des résultats similaires, avec un léger avantage pour F1

- que F1, F2, F3 et F4 sont supérieures à F5 et F6, ces deux dernières faisant intervenir le nombre d'éléments de la classe.

- qu'il n'y a pratiquement pas de différence entre F5 et F6; ceci est certainement dû à l'effet dominant de n_A dans F5.

En conclusion, l'algorithme séquentiel ASC donne de meilleures performances lorsque la fonction-critère est basée uniquement sur une fonction d'homogénéité.

	F1	F2	F3	F4	F5	F6
TCCmax	79.2	78.6	79.1	79.5	73.3	73.3
Ecart-type	8.2	9.8	8.8	8.8	9.9	9.9

Fig.7.10 Moyenne des TCC maximum (en %) et leurs écart-types pour les fonctions F1, F2, F3, F4, F5 et F6 (alg. : ASC , EXP3).

7.6.4 Comparaison de l'algorithme ASC avec l'algortihme UWA

Comme l'algorithme ASC, UWA /74/, /75/ est un algorithme séquentiel, c'est-à-dire qu'il crée les classes l'une après l'autre, et nécessite la fixation à priori d'un seuil de distance.

Soit C' l'ensemble des éléments qui restent à classer et G_0 son centre de gravité. Dans l'algorithme UWA, la première classe-candidate est déterminée par l'ensemble des éléments qui se trouvent à une distance inférieure à T de son centre de gravité G_0. La détermination du centre de gravité G_1 de cette première classe-candidate permet de définir, avec le même critère de seuil, la deuxième classe-candidate. Le centre de gravité G_2 de cette dernière permet de déterminer la troisième classe-candidate, et ainsi de suite. On s'arrête quand deux centres de gravité successifs sont les mêmes, ou lorsque on dépasse un certain nombre (tmax) d'itérations; la k-ième classe est alors la dernière classe-candidate obtenue.

L'algorithme UWA utilisé dans /23/, /74/, /75/ peut être décrit comme suit :

1. Initialisation :

   k = 1 (k-ième classe)
   C' = C

2. a) t = 0 (indice de l'itération)
   b) calculer le centre de gravité de C': G_t
   c) Déterminer la classe-candidate A :

      $A = \{ X_i \in C' \text{ tel que } D(G_t, X_i) < T \}$

e) $t = t + 1$

f) Déterminer le centre de gravité de A : G_t

h) Si ($G_t = G_{t-1}$ ou $t > tmax$) : $C_k = A$ et aller en 3.
Sinon : aller en c)

3. $C' = C' - C_k$

4. Test d'arrêt :
Si $C' \neq \emptyset$: $k = k+1$ et aller en 2.
Sinon : fin

Remarque :

- Le centre de gravité d'un sous-ensemble (C' ou A) est déterminé à l'aide de Minl∞ - ou minmax - (§7.4.3).

Comparaison des TCC pour les algorithmes ASC et UWA

- Nous pouvons dire que l'algorithme UWA est un cas particulier de l'algorithme ASC dans la mesure où le nombre de classes-candidates envisagé par UWA est plus petit que celui envisagé par ASC.

La figure 7.11 donne le taux de classification correcte maximum TCCmax obtenus pour les différents mots par les algorithmes ASC + F2 et UWA (nous avons choisi la fonction F2 égale à minmax(A) dans l'algorithme ASC, car dans l'algorithme UWA le centre de gravité est déterminé par le minmax). On peut constater que la courbe correspondant à ASC + F2 est toujours au-dessus de celle correspondant à UWA. Ainsi, ASC est supérieur à UWA pour tous les mots du vocabulaire choisi.

A titre de comparaison avec les résultats obtenus avec l'algorithme ASC et donnés dans la figure 7.10, on obtient avec l'algorithme UWA un TCCmax moyen égal à 73.5% avec un écart-type de 9.9.

Donc, d'après les tests de classification, où le meilleur partitionnement est en principe connu à priori, l'algorithme ASC (avec une fonction-critère basée sur une fonction d'homogénéité) est supérieur à l'algorithme UWA.

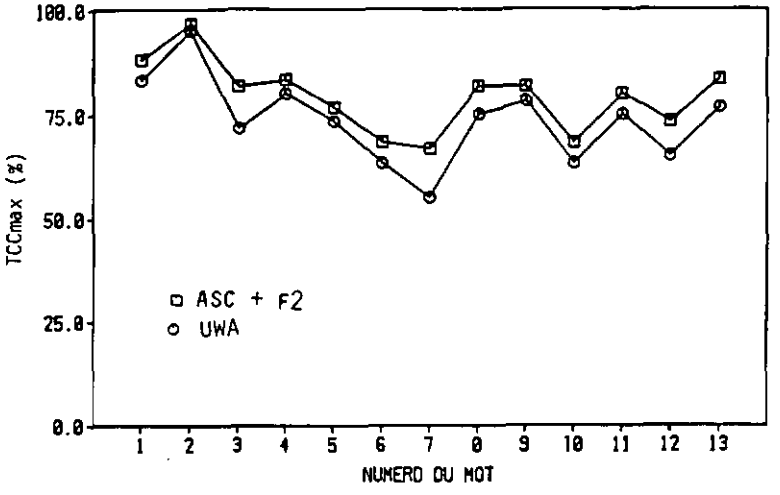


Fig.7.11 TCC maximum pour les algorithmes ASC + F2 et UWA

Nous allons voir ci-après leurs performances respectives en reconnaissance multilocuteur.

7.6.5 Tests de reconnaissance multilocuteur

Les conditions du test sont les mêmes que celles décrites dans §7.5.5

Dans les deux algorithmes ASC et UWA, le seuil de distance pour chacun des mots est choisi de telle façon que l'on obtienne le plus grand nombre de classes, mais au maximum K classes par mot (K valant successivement 6, 5, 4,..., I).

La figure 7.12 donne le taux d'erreur obtenu par les algorithmes ASC avec F1 et UWA.
Les résultats reportés sur cette figure montrent :
- la supériorité de l'algorithme ASC par rapport à l'algorithme UWA.
- comme dans les cas des algorithmes AEC et BI (§7.5), la décroissance du taux d'erreur en fonction du nombre de classes.

Par ailleurs, les résultats obtenus par ASC et UWA sont meilleurs que ceux obtenus avec un choix aléatoire des références (voir fig.7.5).
Nous avons donc montré que l'application de l'algorithme séquentiel basé sur une fonction-critère judicieusement choisie (ASC) pour la création des références multilocuteur permet d'améliorer de façon significative la reconnaissance multilocuteur. En outre, il donne de meilleurs résultats que

l'algorithme UWA /23/,/74/,/75/ précédemment utilisé en reconnaissance multilocuteur.

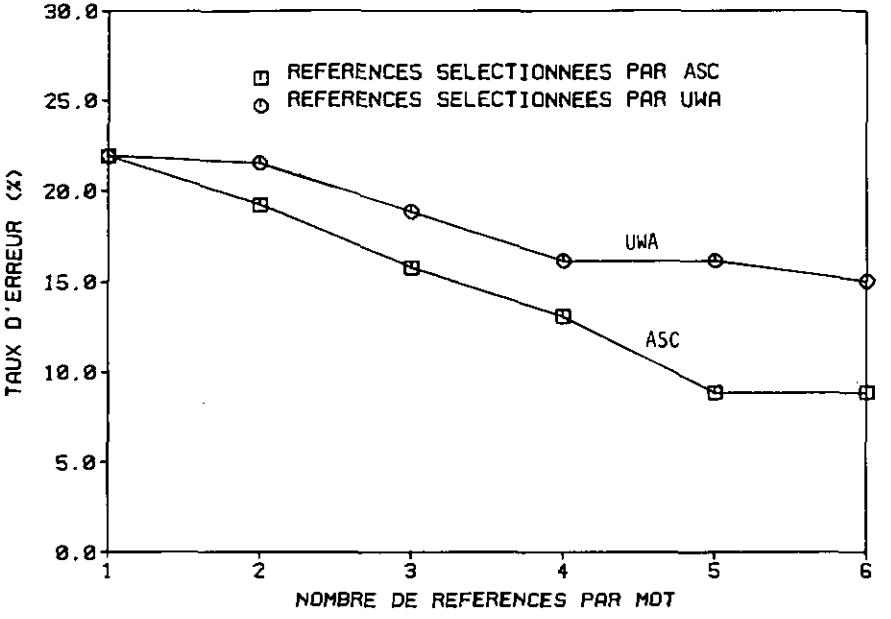


Fig.7.12 Taux d'erreur en reconnaissance multilocuteur en fonction du nombre de références par mot obtenues par les algorithmes ASC et UWA.

7.7 Elimination des isolés

Dans un ensemble de prononciations d'un mot par plusieurs locuteurs, il peut y avoir des 'mauvaises' prononciations. Les causes en sont par exemple :
- un claquement de lèvres au début ou à la fin d'un mot,
- les bruits et les sons produits par l'environnement.

Si la prononciation du mot est perturbée uniquement à son début ou à sa fin, l'algorithme de calcul de la distance entre deux prononciations proposé dans §3.6 permet d'éliminer les effets négatifs que pourrait introduire cette "mauvaise prononciation".

Par contre, si le bruit a lieu 'au milieu' de la prononciation, celle-ci peut perturber de façon significative l'algorithme de classification. Avant d'appliquer la classification automatique pour la création des références multilocuteur, nous devons détecter et éliminer ces "mauvaises prononciations".

Appelons isolé, ou mauvaise prononciation, un élément d'un ensemble qui est éloigné de tous les autres éléments de l'ensemble. La détection des isolés est basée sur la distance d'un élément à son voisin le plus proche qui est grande dans le cas d'un isolé.
Nous admettons que deux 'mauvaises' prononciations sont éloigées l'une de l'autre.

Pour chacun des I éléments (c'est-à-dire les I prononciations d'un mot donné par divers locuteurs), on détermine la distance minimale aux I-1 éléments restants:

$$Dmin(i) = \min_{\substack{j=1,\ldots,I \\ i \neq j}} D(i,j)$$

Un élément i est isolé si :

$$Dmin(i) > \overline{Dmin} + s$$

où $\overline{Dmin}$ est la valeur moyenne des Dmin(i) et s leurs écarts-types.

Les figures 7.13a et 7.13b donnent les résultats de reconnaissance multilocuteur obtenus sans et avec rejet des isolés (selon la méthode proposée ci-dessus). Les remarques suivantes peuvent être faites :

- dans le cas de l'algorithme AEC, les résultats obtenus sans et avec rejet

des isolés sont très proches.

- dans le cas de l'algorithme ASC, les résultats obtenus avec rejet des isolés sont légèrement supérieurs.

Nous pouvons conclure d'après ces résultats, que l'algorithme d'échange AEC est moins perturbé par les isolés que l'algorithme ASC.

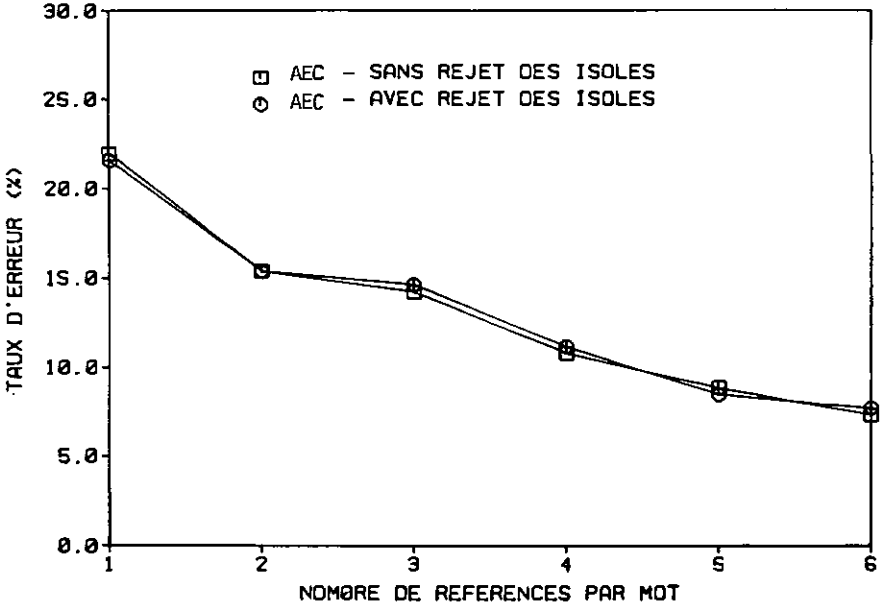


Fig.7.13a

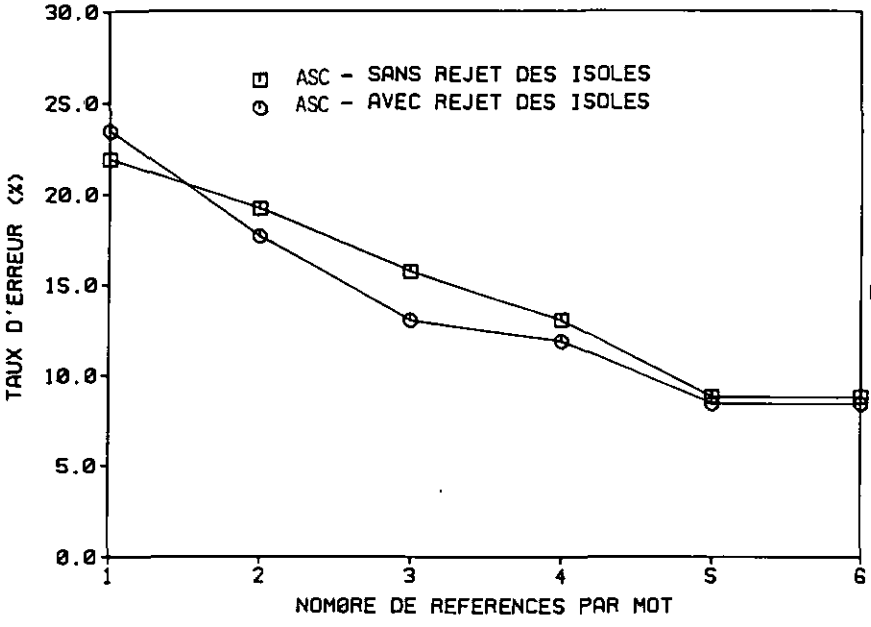


Fig.7.13b

Fig.7.13 Taux d'erreur en reconnaissance multilocuteur en fonction du nombre de références par mot obtenues par les algorithmes :

a) AEC sans et avec rejet des isolés

b) ASC sans et avec rejet des isolés

7.8 Nombre variable de références par mot

Dans un environnement multilocuteur et pour un vocabulaire donné, il existe des mots 'faciles' à reconnaître et d'autres plus 'difficiles' à reconnaître. Si dans un sytème de reconnaissance automatique le nombre total de références pour tout le vocabulaire est fixé, il est plus judicieux d'assigner un nombre plus grand de références pour les mots difficiles à reconnaître, et un nombre plus petit pour les mots faciles à reconnaître.

La figure 7.14 donne la matrice de confusion des mots du vocabulaire comprenant les dix chiffres. Le locuteur-référence est toujours différent du locuteur-test. Le test a été effectué pour une population comprenant 25 hommes et 25 femmes, et où chaque locuteur est pris comme locuteur-référence, les autres constituant les locuteurs-tests.

Nous pouvons constater qu'effectivement, le nombre de confusions, donc d'erreurs, est très variable en fonction du mot. Ce nombre est minimum pour le mot 'six' et maximum pour le mot 'cinq'.

		Référence									
		0	1	2	3	4	5	6	7	8	9
	0	918	36	206	18	0	12	1	33	1	0
	1	31	744	8	385	11	38	1	4	3	0
T	2	170	32	940	3	1	12	2	28	32	5
e	3	27	110	6	1057	8	4	0	12	0	1
s	4	0	13	0	30	896	65	52	117	3	49
t	5	15	41	7	23	28	585	177	341	3	5
	6	0	0	0	0	0	45	1151	22	6	1
	7	13	0	4	2	11	152	413	625	4	1
	8	11	0	49	0	15	23	274	23	665	165
	9	2	0	15	0	75	12	80	17	119	905

Fig.7.14 : Matrice de confusion des dix chiffres avec 25 hommes et 25 femmes. Le locuteur-référence est différent du locuteur-test.

L'idée est d'attribuer à chaque mot un nombre de références proportionnel au taux d'erreur de ce mot, le nombre total des références pour tout le vocabulaire restant inchangé.

Soit Er(m) le taux d'erreur obtenu pour le m-ième mot, Er le taux d'erreur sur tout le vocabulaire et K le nombre moyen de références par mot. Nous fixons le nombre de références K(m) pour le m-ième mot comme suit :

$$K(m) = \frac{Er(m)}{Er} \cdot (K-1) + 1$$

A titre d'exemple, les K(m) qu'on obtient avec les résultats de la figure précédente et avec K = 5 sont les suivants :

m	0	1	2	3	4	5	6	7	8	9
K(m)	4	6	4	3	5	8	2	7	7	4

Ainsi, on affecte au mot 'six' le nombre minimal de classes (K(6) = 2), et au mot 'cinq' le nombre maximal (K(5) = 8).

La figure 7.15 donne les résultats de reconnaissance lorque les références sont sélectionnées à l'aide de l'algorithme ASC dans les cas suivants :

- même nombre de références par mot
- nombre variable de références par mot
- nombre variable de références par mot avec rejet des isolés.

Les résultats obtenus dans le cas où le nombre de références par mot est variable sont meilleurs que le cas où l'on a le même nombre de références par mot.

En fait, l'utilisation d'un nombre variable de références par mot permet d'obtenir les mêmes performances pour tout le vocabulaire, mais avec un nombre total de références plus petit.

Une autre approche intéréssante serait la suivante : en utilisant un algorithme séquentiel et en se basant sur un seuil de distance identique pour tous les mots du vocabulaire, on trouve un nombre de classes variable par mot.

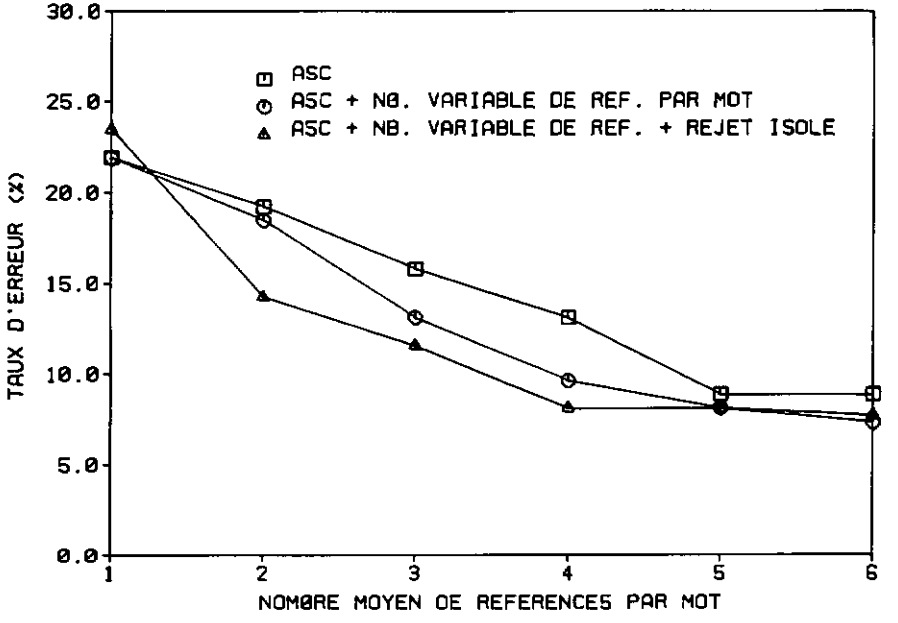


Fig.7.15 Comparaison des résultats en reconnaissance multilocuteur entre un nombre de références fixe par mot et un nombre de références variable par mot.

7.9 Conclusion

Dans ce chapitre, nous avons étudié et mis au point une série de méthodes qui nous permettent, à partir d'un ensemble de prononciations d'un vocabulaire donné par divers locuteurs, de déterminer les groupes typiques pour chaque mot, puis leurs représentants. Ces derniers sont ensuite utilisés comme référence multilocuteur.

Les conclusions importantes sont les suivantes :

1) pour les algorithmes basés sur une fonction-critère, il est important de choisir judicieusement cette fonction pour obtenir des classes utilisables. Pour l'algorithme AEC, les fonctions du type Fb ou Fc sont bien adaptées. Pour l'algorithme ASC, les fonctions-critères basées directement sur une fonction d'homogénéité donnent de meilleurs résultats.

2) Les tests de reconnaissance montrent de façon évidente que les références obtenues à l'aide d'un algorithme de classification donnent de meilleurs résultats que des références choisies de façon aléatoire.

3) Les résultats de reconnaissance sont meilleurs lorsqu'on utilise un algorithme basé sur une fonction-critère (par exemple AEC avec Fb_{11} ou ASC avec F1).

4) Les résultats de reconnaissance obtenus avec AEC sont légèrement meilleurs que ceux obtenus avec ASC.

5) Pour le choix du représentant d'une classe, la méthode moy (moyenne des éléments) donne de meilleurs résultats que les méthodes Minl1 ou $Minl_\infty$. En pratique, la méthode Minl1 qui est légèrement supérieure à $Minl_\infty$ est préférable, car plus simple à mettre en oeuvre.

6) L'utilisation d'un nombre variable de références par mot permet de diminuer le nombre total de références pour tout le vocabulaire, sans détérioration des performances.

Conclusion générale

Les méthodes développées dans ce travail comprennent d'une part, les méthodes de traitement et de compression de données relatives au signal acoustique d'un mot, et d'autre part les méthodes de reconnaissance multilocuteur.

A la suite de l'analyse spectrale à court-terme, les algorithmes de traitement et de compression de données sont décrits comme une suite d'opérations à effectuer : compression temporelle, normalisation en amplitude par zone et quantification. Chaque opération peut être ajustée par l'intermédiaire d'un ou plusieurs paramètres.
Nous montrons, dans le cas d'un petit vocabulaire, que par un choix judicieux des paramètres, on peut réduire considérablement la quantité de données par mot sans dégradation significative des performances.

La reconnaissance multilocuteur a été abordée selon plusieurs approches : le recalage fréquentiel, l'adaptation au locuteur et la création de références multilocuteur.

Les méthodes de recalage fréquentiel permettent surtout d'améliorer la reconnaissance inter-genre. Par contre, aucune amélioration n'est obtenue dans le cas de la reconnaissance intra-genre.

La méthode d'auto-adaptation au locuteur qui est un mécanisme souple d'apprentissage est bien adaptée à des petits vocabulaires et aux systèmes portables.

En ce qui concerne la troisième approche, la partie essentielle de ce travail, nous avons d'abord montré par une analyse factorielle que les prononciations d'un même mot par un grand nombre de locuteurs se regroupent dans un nombre limité de classes.
Nous avons ensuite développé et appliqué deux algorithmes de classification basés sur une fonction-critère (AEC et ASC), en vue de la détermination des références multilocuteur. Nous avons défini toute une série de fonctions-critères et montré celles qui sont bien adaptées au cas de la parole.

Les tests de reconnaissance ont montré l'avantage des algorithmes basés sur une fonction-critère par rapport à d'autres algorithmes déjà appliqués au domaine de la reconnaissance multilocuteur de la parole.
Finalement, nous avons montré que l'utilisation d'un nombre variable de références par mot permet de réduire le nombre total de références pour tout le vocabulaire sans détérioration des performances.

Annexe A

Paramètres du traitement numérique et du calcul de distance

Normalisation temporelle (NTMP) :

NTMP =	1	Normalisation linéaire
	2	Normalisation curviligne
N		Longueur de normalisation

Normalisation en amplitude (NAMP) :

NAMP =	Z0	Norm. en amplitude globale
	Z1	Norm. en amplitude par ligne
	Z2	Norm. en amplitude par demi-colonne
	Z3	Norm. en amplitude par colonne
	Z4(LF,LT)	Norm. en amplitude par zone de longueur fréquentielle LF et de longueur temporelle LT

Quantification (QUANT) :

QUANT =	1	Quantification linéaire
	2	Quantification logarithmique
Nb		Nombre de bits
r0		Valeur, dans la matrice non quantifiée, à laquelle correspond la valeur quantifiée nulle. r0 est rapporté à la valeur moyenne de la matrice non quantifiée.
r1		Valeur, dans la matrice non quantifiée, à laquelle correspond la valeur quantifiée maximale. r0 est rapporté à la valeur moyenne de la matrice non quantifiée.

Distances entre deux matrices (DMAT) :

DMAT =	1	Distance linéaire simple
	2	Distance linéaire avec déplacement global
	3	Distance dynamique
DVEC =	1	Distance entre vecteurs-fréquences selon la relation (3.1)
	2	Distance entre vecteurs-fréquences selon la relation (3.2)
RANGE		Largeur de la fenêtre dans le cas de la distance dynamique, ou déplacement global maximum dans le cas de la distance linéaire.
COF =	1	On impose que les débuts et fins de deux mots à comparer coïncident (cas de la distance dynamique).
	0	Pas de contraintes.

Annexe B

Glossaire

signal acoustique : forme électrique du son à la sortie du microphone.

analyse acoustique : caractérisation de la variation temporelle du spectre d'amplitude.

cognitif : ayant trait à la signification du langage et plus généralement à la connaissance du monde.

classe : ensemble d'éléments ayant des traits similaires. Un mot peut être décrit par une seule classe dans le cas d'un système monolocuteur et par plusieurs classes dans le cas d'un système multilocuteur.

diphonème : association de deux phonèmes.

fondamental : fréquence correspondant à la vibration des cordes vocales.

formant : fréquence correspondant à une résonnance du conduit vocal.

lexique : les mots du vocabulaire d'une langue.

locuteur : le locuteur est le sujet qui parle.

locuteur-test : locuteur à qui appartient les mots-tests.

locuteur-référence : locuteur à qui appartient les mots-références.

matrice brute : représentation matricielle du signal à la sortie de

l'analyseur acoustique.

mot : son ou groupe de sons.

mot isolé : mot prononcé de façon isolé.

mots enchaînés : séquence isolée de quelques mots prononcés de façon continue. Le nombre maximum de mots est fixé et petit.

mot-référence : forme traitée d'un mot se trouvant dans le dictionnaire.

mot-test : forme traitée d'un mot inconnu

phonème : son élémentaire d'un langage (en français il y en a une trentaine).

pragmatique : étude des corrélations des formes linguistiques et des variables situationnelles.

prosodie : concerne la mélodie, le rythme et l'intensité de la voix.

sémantique : étude des mots considérés dans leur signification.

syntaxe : partie de la grammaire qui traite de la fonction, la disposition des mots et des propositions dans la phrase.

vecteur-fréquence : ensemble des valeurs de sortie des K canaux de l'analyseur spectral à un instant d'échantillonnage.

BIBLIOGRAPHIE

1. Allen J.B. and Rabiner L.R., "A unified approach to short-time Fourier analysis and synthesis", Proc. IEEE, vol. 65 no. 11, pp. 1558-1564, 1977.

2. Arbenz K., Wohlhauser A., "Analyse numérique", Presses polytechniques romandes, Lausanne, 1980.

3. Baker J.M., "A new time-domain analysis of human speech and other complex waveforms", Ph.D., Carnegie-Mellon University, Pittsburg, PA., 1975.

4. Ball G.H. and Hall D.J., "Isodata - An Iterative Method of Multivariate analysis and Pattern Classification", in Proc.IFIPS Congr., 1965.

5. Baker J.K., "The DRAGON system : an overview", IEEE Transactions, vol. ASSP-23, pp. 24-29, 1975.

6. Bedrosian E., "A product theorem for Hilbert transforms", Proceedings of the IEEE, pp. 868-869, 1963.

7. Bellman R. and Dreyfus S.E., "Applied Dynamic Programming", Princeton University Press.

8. Benzecri J.P., "L'analyse des données : la taxinomie", Dunod, Paris, 1980.

9. Bui N. C., Monbaron J.J., and Michel J.M., "An Integrad Voice Recognition System", IEEE Trans. on ASSP, Vol.ASSP-31 No. 1, February 1983.

10. Bouroche J.M. et Saporta G., "L'analyse des données", Presses

Universitaires de France, 1980.

11. Billi R., Oreglia M., Pieraccini R., Scagliola, Vicenzi C., "Performance analysis of speaker-trained isolated word recognition system", Int. Zurich seminar, 1982.

12. Oeczky A.G., "Recognition of isolated words for small vocabularies", Rep. No. 38 EC 02 001, IMT Neuchâtel, August 1980.

13. Oeczky A.G., "Recognition of isolated words for small vocabularies", Rep. No. 38 EC 02 002, IMT Neuchâtel, october 1980.

14. De Mori R., "Recent Advances in Automatic Speech Recognition", ICPR, Kyoto, Japan, Nov. 1978.

15. Diday E., Lemaire J., Pouget J., Testu F., "Eléments d'analyse de données", Dunod, Paris, 1982.

16. Diday E. et Coll. "Optimisation en classification automatique", INRIA, Rocquencourt, France, 1980.

17. Duda R.O., Hart P.E., "Pattern classification and scene analysis", John Willey, New York, 1973.

18. Erman L.D., "A functional description of the HEARSEAY II system", Proc. IEEE-ICASSP, Hartford, Conn., pp. 799-802, 1977.

19. Eckart C., Young G., "The approximation of one matrix by another of lower rank", Psycometrika, Vol. 1 No 3, september 1936.

20. Elenius K., Blomberg M., "Effects of emphasizing transitional or stationnary parts of the speech signal in a discrete utterance recognition system", IEEE Proceedings of ICASSP, Vol.1, pp. 535-538, 1982.

21. Fant G., "Speech communications", Almqvist and Wiksell, Stockholm.

22. Flanagan James L., "Speech analysis, synthesis, and perception", Springer-Verlag, New York, 1972.

23. Flocon B. and Lockwood P., "A Speaker Independent Isolated Word Recognition System", EUSIPCO-83, pp. 407-410, 1983.

24. Fu K.S., "Digital pattern recognition", Springer-Verlag, New York, 1980.

25. Fu K.S., "Applications of Pattern Recognition", CRC Press, Palm Beach, 1982.

26. Grenier Y., Maurin J.C. "Adaptation au locuteur par analyse canonique des corrélations", 2ème congrès Reconnaissance des formes et Intelligence Artificielle", Toulouse, 1979.

27. Gauvain J.L., Mariani J., Lienard J.S., "On the use of time compression for Word-based recognition", ICASSP-83, Boston, 1983.

28. Gauvin J.L., Mariani J., "A method for connected word and word spotting recognition", IEEE Proceedings of ICASSP-82, Vol.2, pp. 891-894, 1982.

29. Harris F.J., "On the use of windows for harmonic analysis with the discrete Fourier transform", Proc. IEEE, vol. 66, no. 1, pp. 51-83, 1978.

30. Haton J.P. "Contribution à l'analyse, la paramétrisation et la reconnaissance automatique de la parole", Thèse d'état, Nancy, 1974.

31. Haton J.P., "Reconnaissance automatique de la parole", 8-ième école d'été d'informatique de l'AFCET, Namur 1978.

32. Haton J.P., "Speech recognition and understanding", 6th ICPR, Munich, 1982.

33. Haton J.P., "Automatic speech analysis and recognition", D.Reidel publ., Oordrecht, 1982.

34. Haton J.P., "Reconnaissance automatique de la parole : état de la recherche et du développement", dans "Le point sur la reconnaissance et la synthèse de la parole", Agence de l'informatique, Paris, La défense, 1981.

35. Hugli H. et Mokeddem A., "Reconnaissance du locuteur et de mots isolés par des systèmes miniaturisés : une comparaison", Journées d'électronique 85, real time processing : application in control and signal processing, Lausanne, Suisse, Oct. 1985.

36. Itakura F., "Minimum prediction residual principle applied to speech recognition", IEEE Trans. on ASSP-23, Vol. No.1, February 1975.

37. Jelinek F., "Continuous speech recognition by statistical method", Proceedings IEEE, vol. 64, no. 4, pp. 532-556, 1976.

38. Jaschul J., "Speaker adaptation by a linear transformation with optimised parameters", Proceedings of ICASSP 82, Paris, 1982.

39. Klatt D.H., "Review of the ARPA speech understanding project", J.A.S.A., vol. 62, pp. 1345-1366., 1977.

40. Knuth Donald E., "The art of computer programming", Addison-Wesley publishing company, Vol. 1.

41. Kunt M., "Traitement numérique des signaux", Editions Georgi, Lausanne, 1980.

42. Lamel L.F., Rabiner L.R., Rosenberg A.E. and Wilpson J.G., "An Improved endpoint detector for IWR", ASSP-29, 1981.

43. Lea W.A., "Trends in Automatic Speech Recognition", Englewood Cliffs, Prentice-Hall, New-York, 1980.

44. Lea W.A., "Selecting, Desigining, and Using Practical Speech Recognizers" in "Automatic Speech Analysis and Recognition" by J.P. Haton, D.Reidel Publishing Compagny, 1982.

45. Lea W.A., "Selecting, Desigining, and Using Speech Recognizers". Santa Barbara, CA: Speech Science Publications, 1982.

46. Lebart L., Morineau A., Fénelon J.P. " Traitement des données statistiques", Dunod, 1982.

47. Levinson S.E., Rabiner L.R., Rosenberg A.E, and Wilpson J.G., "Interactive Clustering Techniques for Selecting Speaker-Independent Reference Templates for Isolated Word Recognition", IEEE Trans. on ASSP, Vol. ASSP-27, No.2, April 1979.

48. Levinson S. et Liberman M., "La reconnaissance de la parole par ordinateur", dans "L'intelligence de l'informatique", bibliothèque pour la science, diffusion Belin, 1984.

49. Liénard J.S., "Les processus de la communication parlée", Masson, Paris, 1977

50. Lowerre, B.T., "The HARPY speech recognition system", Ph.D., Carnegie-Mellon University, Pittsburgh, Pa., 1976.

51. Lowerre, B.T., "Dynamic Speaker adaptation in the HARPY Speech Recognition System", Int. Conf. ASSP, New-York, 1977.

52. Makhoul J., Viswanathan R., Schwartz R., Huggins A.W.F., "A mixed-source model for speech comprehension and synthesis", Proc. IEEE-ICASSP, Tulsa, 1978.

53. MacQueen, "Some methods for classification and analysis of multivariate data", in Proc. 5th Berkeley Symp. Probability and statistics, Berkeley, CA, 1967.

54. Makhoul J., "linear prediction : a tutorial review", Proceedings IEEE, vol. 63, pp. 561-580, 1975.

55. Markel J.D. and Gray Jr. A.H., "Linear Prediction of Speech", Springer-Verlag, Berlin, 1976.

56. Martin, T.B., "Practical applications of voice input to machine", Procedings IEEE, vol. 64, no. 4, pp. 487-501, 1976.

57. Minoux M. "La programmation mathématique", théorie et algorithmes (tomes 1 et 2), Paris, 1983.

58. Mokeddem A., "Reconnaissance de la parole: algorithmes de traitement

numérique", rapport 99 EC 02 007, IMT Neuchâtel.

59. Mokeddem A., "Création d'une référence indépendante du locuteur pour la reconnaissance automatique des mots isolés", rapport 53 EC 02 0028, IMT Neuchâtel, 1981.

60. Mokeddem A., "Reconnaissance multilocuteur des mots isolés par recalage fréquentiel", rapport 115 EC 02 009, IMT Neuchâtel.

61. Mokeddem A., "Analyse factorielle appliquée aux échantillons multilocuteur de la parole", rapport 116 IC 02 010, IMT Neuchâtel.

62. Mokeddem A., "Techniques de classification automatique appliquées à la reconnaissance multilocuteur", rapport 114 EC 02 008, IMT Neuchâtel.

63. Mokeddem A., "Méthodes statistiques appliquées à la reconnaissance multilocuteur de mots isolés", rapport 156 EC 03/85, IMT Neuchâtel.

64. Mokeddem A., Hügli H., Pellandini F., "Evaluation of criterion based clustering procedures for generating multiple template references in speaker independent speech recognition", seventh ICPR, Montreal, August 1984.

65. Mokeddem A., Hügli H., Pellandini F., "Criterion based clustering techniques applied to speaker independent speech recognition", conference on "Digital Processing of Signals on Communications", at University of Loughbrough, England 22-25 April 1985.

66. Mokeddem A., Hügli H., Pellandini F., "Reconnaissance multilocuteur de mots isolés", 14-ièmes journées d'étude sur la parole, GALF, Paris, Juin 1985.

67. Niles Les, Silverman Harvey F., Oixon R., "A Comparison of Three Feature Vector Clustering Procedures in a Speech Recognition Paradigm". Proc. ICASSP 83, pp.765-568, 1983.

68. Oppenheim A.V. and Shafer R.W., "Digital signal processing", Prentice Hall, Englewood Cliffs, NJ, 1975.

69. Okochi M., Sakai T., "Trapezoïdal DP Matching with Time Reversibility", Proc. ICASSP 82, pp. 1239-1242, 1982.

70. Pieraccini R., "Pattern compression in isolated isolated word recognition", Signal processing, Volume 7, No. 1, Septembre 1984.

71. Rabiner L.R. and Sambur M.R., "An algorithm for determining the endpoints of isolated utterances", Bell System Technical Journal, vol. 54, no. 2, pp. 297-315., 1975.

72. Rabiner L.R., Rosenberg A.E. and Levinson S.E., "Considerations in dynamic time warping for discrete word recognition", IEEE Transactions, vol. ASSP-25, no. 4, pp. 338-345.

73. Rabiner L.R., "Digital processing of speech signals", Prentice-Hall, New Jersey, 1978.

74. Rabiner L.R., "*On Creating References for Speaker Independent* Recognition of Isolated Words", IEEE Trans on ASSP, Vol. ASSP, No.3, pp.34-42, February 1978.

75. Rabiner L.R., Wilpson J.G., "Considerations in Applying Techniques to Speaker-independent Word Recognition", J. Acoust. Soc. Am., Vol. 66, No. 3, September 1979.

76. Renaud M., "Réseau de processeurs pour l'algorithme DTW en reconnaissance de mots isolés", CEH Neuchâtel rapport technique No. 264, janvier 1983.

77. Sakoe H., "Two level dp-matching - A dynamic programming based pattern matching algorithm for connected word recognition", IEEE transactions, vol. ASSP-27, no. 6, pp. 588-595, 1979.

78. Sakoe H., Chiba S., "Dynamic Programming Algorithm Optimization for Spoken Word Recognition", IEEE Trans on ASSP, Vol. ASSP-26 No. 1, pp. 43-49, February 1978.

79. Sambur M.R. and Rabiner L.R., "A speaker independent digit recognition system"", Bell Syst. Tech. J., vol.54, Jan. 1975.

80. Schafer R.W. and Markel J.O., "Speech Analysis", IEEE Press, New-York, 1979.

81. Suen C.Y. and De Mori R., "Computer analysis and perception of visual and auditory signals", CRC press, Roton Boca, Fla., U.S.A., 1981.

82. Tou J.T. and Gonzales, R.C. "Pattern Recognition Principles", Addison-Wesley, 1974.

83. Tsuruta S., "OP-100 voice recognition system achives high efficiency", JEE, pp. 50-54, 1978.

84. Velichko V.M., Zagorujko N.G., "Automatic recognition of 200 words", Int. J. Man-Machine Studies, 2, 1970.

85. Wilpon J.G., Rabiner L.R., "A Modified K-means Clustering Algorithm for Use in Isolated Work Recognition", IEEE Transactions on ASSP, Vol. ASSP-33, no 3, June 1985.

86. Young G. and Householder A.S., "Discussion of a set of points in terms of their mutual distances", Psychometrika.

87. Zoïcas Adrian, "Reconnaissance vocale : un jeu de circuits intégrés mis au point par NEC", Minis et Micros no 215., 1985.

Remerciements

Cette étude a été réalisée à l'Institut de Microtechnique de l'Université de Neuchâtel (Suisse) sous la direction du Professeur F. Pellandini. Je tiens à lui exprimer ma profonde reconnaissance pour l'intérêt permanent qu'il a porté à ce travail et les conseils et encouragements qu'il n'a cessé de me prodiguer.

J'exprime également mes remerciements au Dr. H. Hugli, membre du jury de thèse, pour les nombreuses discussions que nous avons eues, pour ses nombreuses remarques et pour ses suggestions lors la rédaction de ce document.

Pour ses conseils, ses nombreuses remarques et encouragements, je remercie chaleureusement le Professeur M. Kunt du Laboratoire de traitement des signaux de l'Ecole Polytechnique Fédérale de Lausanne (EPFL), également membre du Jury de thèse.

Mes remerciements vont également à tous mes collègues de l'IMT pour leur collaboration, ainsi qu'à toutes les nombreuses personnes qui ont accepté d'être enregistrées.

J'exprime, enfin, ma gratitude à Mlle S. De cerjat pour ses conseils lors de la rédaction de ce document.

Une partie du présent travail de recherche a bénéficié du soutien de la "Commision pour l'Encouragement de la Recherche Scientifique" (CERS no 1158, Berne, Suisse) et des entreprises suivantes : ASULAB SA, CSEM SA, METTLER SA, HASLER SA, AUTOPHON SA et CIR SA.