الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم و البحث العلمي
MINISTERE DE L'ENSEIGNEMENT ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

RECONNAISSANCE De La PAROLE Par La METHODE GLOBALE

Proposé par :
Mr. Boualem BOUSSEKSOU

Etudié par
Ferhat SALAH
Leila SAHLI

Dirigé par :
Mr. Boualem BOUSSEKSOU

Promotion : Janvier 1986

E.N.P 10, Avenue Hacen Badi — EL-HARRACH — ALGER

RECONNAISSANCE DE LA PAROLE
PAR LA
METHODE GLOBALE

DEDICACES

A ma mère
A tous ceux qui me sont chers

Leïla .

* * *

A mon père , à ma mère pour leurs sacrifices
A mon frère Mahmoud
A mes frères et soeurs
A toute ma famille
A mes amis(es)

Ferhat .

* * * * * * * *

REMERCIEMENTS

Nous tenons à exprimer nos plus vifs remerciements à notre promoteur Mr B.BOUSSEKSOU , pour son aide et ses conseils tout au long de l'élaboration de notre projet de fin d'études.

Nous remercions également Mr A.MENACER , Mr C.DJESSAS, Mr B.SAHLI et Mr M.SALAH , pour leur aide qui nous a été très précieuse .

Nos remerciements vont aussi à Melle A.MOUSSAOUI,pour la documentation,très utile,qu'elle nous a fournie .

Nous n'oublions pas Mr R.BOUKECHOUIOU , que nous tenons à remercier , pour nous avoir aidé dans l'exécution de ce document .

Que tous ceux qui ont contribué à notre formation , trouvent ici l'expression de notre profonde gratitude .

TABLE DES MATIERES

* * * * * * * * * * * * * *

I. GENERALITES

I.1 Introduction

Construire des machines qui parlent et obéissent à la voix humaine,c'est un vieux rêve que l'homme est en train de réaliser.Le meilleur(et vieux)moyen de communication,qu'est la parole,devient peu à peu un nouvel outil.En ce qui nous concerne,il faut dire que ce domaine est beaucoup plus vaste qu'on le pense pour pouvoir être traité en entier.Et dans les pages qui vont suivre ,nous ne lirons que sur la reconnaissance de la parole par la méthode globale.Elle est basée sur une adaptation non linéaire de deux mots.Cette adaptation non linéaire est dûe au fait que la caractéristique la plus évidente du signal de la parole est sa très grande variabilité.On a alors recours à la technique de programmation dynamique.

Avant la phase de reconnaissance proprement dite,la machine doit mémoriser la prononciation du locuteur(sous forme de paramètres pertinents fournis,au préalable,par l'analyse),l'empreinte de la voix de son maître.Cette phase,dite d'apprentissage,correspond à la création des données de références associées au vocabulaire du locuteur,données aussitôt stockées en mémoire.Enfin dans la phase qui nous intéresse,il y a d;abord mémorisation des paramètres d'analyse, du mot entendu,puis la machine les compare à ceux de tous les mots qu'elle trouve dans son dictionnaire.Les mots les plus proches correspondant aux distances calculées les plus faibles,sont supposés les plus probables.

Cependant la reconnaissance de la parole ne va pas sans rencontrer de difficultés:Lorsque nous parlons,l'information est très redondante.La prononciation varie d'une personne à une autre(forme du conduit vocal,âge,sexe,origine géographique...)et, la plupart des mots d'une phrase étant enchainés,la machine a le plus grand mal à distinguer les mots dans une phrase continue(problème de segmentation).Ces problèmes disparaissent si on se limite à une reconnaissance de mots isolés,en mode monolocuteur,même si chacun d'entre nous ne prononce pas toujours un même mot de la même façon.

Dans ce rapport,nous présentons essentiellement les algorithmes proposés par MYERSet RABINER pour la reconnaissance du mot connecté (qui peut être un fragment de phrase).Ces deux chercheurs utilisent la méthode de construction par niveaux(Level Building)afin justement de résoudre le problème de segmentation.L'algorithme de LB final est accompagné de plusieurs modifications destinées à l'optimiser.Nous avons alors proposé un algorithme LB modifié.

La priorité est donnée à la méthode utilisée pour la reconnaissance.De ce fait,la phonétique et l'analyse,loin de diminuer de leur importance certaine,ne sont que brièvement exposées.

I.2 Les sons

Il n'est pas inutile de parler, même de façon limitée, des sons. On appelle "son", tout message naturel ou provoqué perçu par l'inter-médiaire de l'ouie. Il se caractérise par sa fréquence, son intensité et son timbre.

Complètement fermées, les cordes vocales vibrent à la manière d'une anche double, et la voix est dite "voisée" ; lorsqu'elles sont entièrement ouvertes, le son produit est un bruit d'écoulement dont le spectre est similaire à celui d'un bruit blanc, et la voix est non voisée / 2 / . Les voyelles sont, par exemple, produites par excitation des cordes vocales.

Le conduit vocal peut engendrer les consonnes fricatives dont la prononciation se caractérise par un frottement de l'air contre les dents ou les lèvres telles que f, v, s, z, j, ch. Ces sons peuvent être voisés ou non voisés selon qu'ils sont accompagnés, ou non, de la vibration des cordes vocales.

Il est aussi à l'origine de bruits d'explosions qui proviennent de l'occlusion momentanée du conduit vocal, suivie d'une brusque ouverture : ainsi obtient-on les plosives qui sont voisées (sons "be", "de", "gue"), ou non voisées (sons "pe", "te", "ke").

Enfin le nez est mis à contribution dans les consonnes nasales comme m ou n, ou par couplage du conduit nasal et du conduit buccal (sons "on", "an", "in", produits bouche ouverte).

II. ANALYSE

Le but de l'analyse est l'extraction des paramètres pertinents caractérisant le signal vocal.Elle suppose que la parole a été délimitée(détection bruit/parole) /8/ ,/1/ .

II.1 Prétraitement

Avant d'être analysé,le signal de parole doit d'abord subir un prétraitement que constituent le fenêtrage et la préaccentuation.

Fenêtrage: Son rôle est de limiter la durée d'un signal.La fenêtre communément choisie pour le signal de la parole est la fenêtre de Hamming /7/ .

Préaccentuation: Une préaccentuation numérique de 6 dB/octave compense les effets dûs à la source d'excitation du conduit vocal (-12 dB/octave)et au rayonnement des lèvres(+6 dB/octave)sur certains sons voisés.Cette opération permet donc de rétablir le signal de parole tel qu'il était avant de sortir de la bouche.

II.2 Méthodes d'analyse

Les méthodes d'analyse sont diverses.Pour des raisons indiquées ci-après,notre choix s'est porté sur la méthode cepstrale. Aussi nous ne nous attarderons pas sur les autres.Néanmoins on peut citer:

Analyse spectrale:

*type analogique: méthode par filtrage(vocodeur à canaux, à formants) /5/ .

*type numérique: FFT /1/ .

Analyse par prédiction linéaire:L'idée principale de cette méthode est que les échantillons de parole peuvent être approximés par une combinaison linéaire des échantillons de parole les précédant.Le calcul des coefficients de PL se fait en minimisant l'erreur quadratique entre le signal original et la valeur estimée par combinaison linéaire.La résolution peut se faire de différentes manières,notamment la méthode de covariance et la méthode d'auto-corrélation /15/ .

II.3 Analyse cepstrale

Pourquoi l'analyse cepstrale ?

Elle a paru intéressante en reconnaissance grâce à son économie de représentation(peu de paramètres:8 coefficients au lieu 12pour la méthode PL et 20 pour la méthode spectrale)/4/ et à son intérpre- -tation physique en termes de distance(comparaisons entre paramètres)

Son but est de réaliser une séparation entre l'excitation et l'enveloppe spectrale du conduit vocal.

Un segment de parole peut être décrit par une convolution com--binant les effets dûs à la source,au conduit vocal et aux lèvres. La déconvolution a pour but de séparer ces éléments de convolution, lequel but peut être atteint par le biais du traitement homomorphique. Le moyen utilisé est le cepstre/7/ /8/ /13/ .

L'opération de déconvolution vérifie l'équation:

$$D(x(n)) = D(x1(n)*x2(n)) = D(x1(n)) + D(x2(n))$$

D étant l'opérateur de déconvolution.

On sait que la transformée en z d'un produit de convolution est le produit des transformées en z des termes qui le constituent,c-à-d:

$$X(z) = X1(z) \cdot X2(z)$$

ce produit peut être transformé en somme à l'aide de la fonction logarithme.On a:

$$\hat{X}(z) = \log X(z) = \log(X1(z) \cdot X2(z))$$

$$\hat{X}(z) = \log X1(z) + \log X2(z)$$

Le logarithme utilisé est le logarithme complexe,c-à-d :

$$\hat{X}(z) = \hat{X}(e^{jw}) = \log|X(e^{jw})| + j \arg X(e^{jw})$$

Le cepstre complexe,$\hat{x}(n)$,d'un signal est donné par la transformée de Fourier inverse du log de la transformée de Fourier(ou spectre) de ce signal.

$$\hat{x}(n) = 1/2\pi \int_{-\pi}^{\pi} \hat{X}(e^{jw}) \, e^{jwn} \, dw$$

avec $\hat{X}(e^{jw}) = \log X(e^{jw})$.

Le cepstre, qui nous intéresse, est donné par :

$$c(n) = 1/2\pi \int_{-\pi}^{\pi} \log/X(e^{jw})/\ e^{jwn} dw \qquad -\infty < n < \infty$$

Il peut être aussi défini par :

$$c(n) = (\hat{x}(n) + \hat{x}(-n))/2 \quad \text{c-à-d :}$$

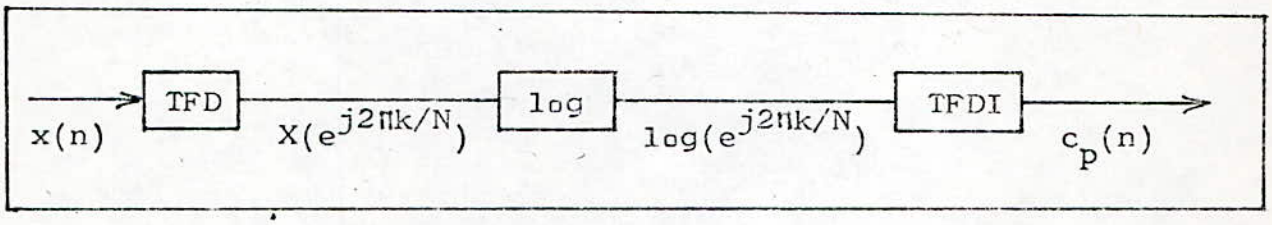
$$c(n) = 1/2\pi \int_{-\pi}^{\pi} \log(X(e^{jw}))\ \cos(wn)\ dw\ .$$

Le cepstre peut donc être obtenu de deux manières :

* soit par deux transformations de Fourier .
* soit par une transformation en cosinus du log du spectre .

En pratique, on utilise une approximation pour éviter le calcul intégral : on calcule l'inverse de la transformée de Fourier discrète (TFDI) du log de la TFD du signal d'entrée, c-à-d :

$$c_p(n) = 1/N \sum_{k=0}^{N-1} \log/x_p(k)/\ e^{j2\pi kn/N} \qquad \text{pour} \quad 0 \leqslant n \leqslant N-1$$



III. COMPARAISON DYNAMIQUE DES MOTS

III. 1 Programmation dynamique

Toute comparaison entre mots aura besoin d'une adaptation non linéaire : Un même mot prononcé par une seule personne voit sa durée varier suivant le contexte, l'intonation et l'état de la personne. Les différences de durée peuvent provoquer un décalage important entre les diverses parties du mot et celles de l'empreinte (mot de référence). On peut alors étirer ou comprimer l'empreinte de façon à minimiser une mesure globale de l'écart entre elle et le mot prononcé.

La programmation dynamique est une technique qui réalise cette adaptation. Elle est basée sur le principe d'optimalité énoncé par Richard BELLMAN en 1957 et qui peut être resumé par : Toute sous-stratégie d'une stratégie optimale est elle même optimale /3/ .

III.2 Le principe d'adaptation par programmation dynamique

III.2.1 Définition de la distance normalisée dans le temps

La parole peut être exprimée par une séquence de vecteurs constitués de paramètres pertinents (par exemple les coefficients ceptraux).

Soient deux blocs de parole A et B tel que :

$$A = a1, a2, \ldots, ai, \ldots, a_I$$

$$B = b1, b2, \ldots, bj, \ldots, b_J \qquad \text{(III.1)}$$

La comparaison entre ces deux blocs de parole, basée sur une adaptation non linéaire, se fait comme suit :
on considère un plan (i,j) (fig. III.1) où A et B occupent respectivement les axes i et j. Les différences d'ajustement entre eux peuvent être décrites par une séquence de points c (i,j).

$$F = c(1), c(2), \ldots, c(k), \ldots, c(K).$$

où

$$c(k) = (i(k), j(k)) \qquad \text{(III.2)}$$

Cette séquence constitue une fonction qui réalise une super-position des deux blocs de parole. Elle est appelée fonction de déformation (ou de coincidence). Quand il n'y a pas de différence d'ajustement entre ces deux formes, la fonction de déformation coincide avec la diagonale j = i.

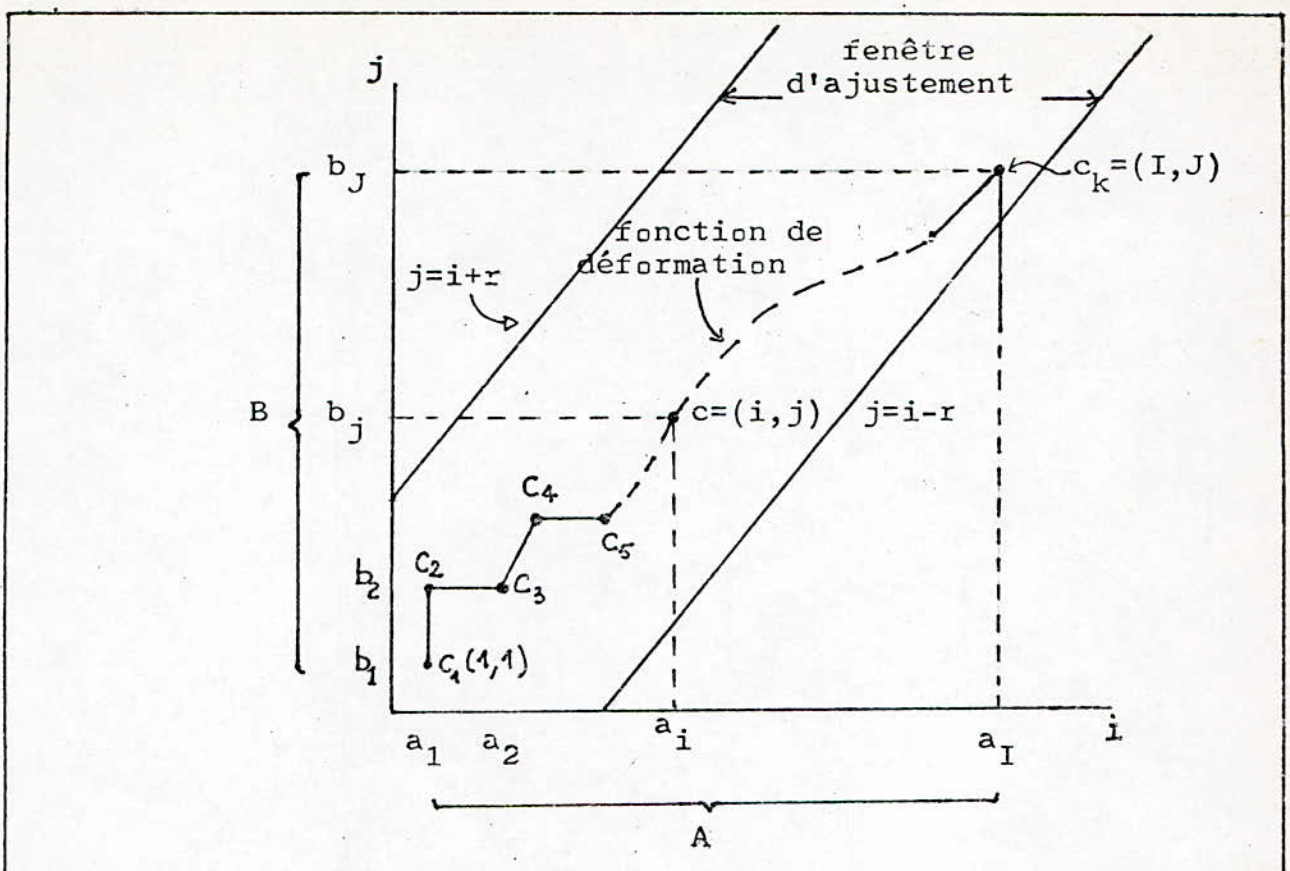


Fig.III1 Fonction de déformation et fenêtre d'ajustement.

Pour évaluer la différence entre deux vecteurs ai et bj, la distance utilisée est :

$$d(c) = d(i,j) = \|ai - bj\| \qquad \text{(III.3)}$$

Remarque sur le choix de la distance :

Afin de simplifier les calculs et dans le cas des coefficients cepstraux, on utilise l'écart absolu entre eux.

$$d(c) = /ai-bj/.$$

La sommation pondérée des distances sur la fonction de déformation F donne :

$$E(F) = \sum_{k=1}^{K} d(c(k)).\ w(k) \qquad \text{(III.4)}$$

où w(k) est un coefficient de pondération, non négatif, introduit afin de permettre une certaine flexibilité dans la mesure de E(F). E(F) atteind son minimum quand F décrit le meilleur ajustement des deux blocs. Cette valeur de distance minimum peut être considérée

comme étant une distance entre A et B, et on s'attend à ce qu'elle soit stable malgré les fluctuations dans le temps.

Basée sur ces considérations, la distance normalisée entre A et B, est définie comme suit :

$$D(A,B) = \underset{F}{\text{Min}} \left[\frac{\sum_{k=1}^{K} d(c(k)) \cdot w(k)}{\sum_{k=1}^{K} w(k)} \right] \qquad \text{(III.5)}$$

où $\sum_{k=1}^{K} w(k)$ est employé pour compenser l'effet de K (nombre de points sur la fonction de déformation F).

Cette grandeur, D(A,B), dépend essentiellement de la fonction de déformation et de la définition des coefficients de pondération. Dans notre cas, le problème est restreint au cas le plus général où les deux conditions suivantes sont réalisées.

1ère condition : les blocs de parole sont échantillonnés avec la même période d'échantillonnage.

2ème condition : nous ne connaissons pas, à priori, les parties de la parole qui contiennent les informations linguistiques les plus importantes. Il est donc raisonnable de supposer que chaque partie de la parole contient la même quantité d'information linguistique.

III.2.2 Restrictions sur la fonction de déformation

La fontion F définie comme étant la superposition de deux blocs de parole A et B, doit préserver les structures linguistiques essentielles des axes portant A et B notamment : la continuité, la monotonie et la limitation de la vitesse de transition des paramètres acoustiques. Ces conditions peuvent être satisfaites en imposant les restrictions suivantes sur F(ou sur les points $c(k) = (i(k), j(k))$) :

1. condition de monotonie

$$i(k-1) \leqslant i(k) \quad \text{et} \quad j(k-1) \leqslant j(k)$$

2. condition de continuité

$$i(k) - i(k-1) \leqslant 1$$

et

$$j(k) - j(k-1) \leqslant 1$$

Ces deux restrictions conduisent à la relation que doit satisfaire deux points consécutifs :

$$c(k-1) = \begin{cases} (i(k),\ j(k)-1) \\ (i(k)-1,\ j(k)-1) \\ \text{ou } (i(k)-1,\ j(k)) \end{cases} \qquad \text{(III.6)}$$

3.Conditions aux limites

$$i(1) = 1, \quad j(1) = 1$$
$$i(K) = I, \quad j(K) = J \qquad \text{(III.7)}$$

4.condition d'ajustement (ou fenêtre d'ajustement) (fig. III.1)

$$/i(k) - j(k)/ \leqslant r \qquad \text{(III.8)}$$

où r est un nombre entier positif appelé longueur de la fenêtre. Cette condition correspond au fait que les fluctuations ne causent jamais une différence d'ajustement trop excessive.

5. contrainte sur la pente

Une pente trop raide ou trop"douce" causerait des déformations indésirables. Trop raide, par exemple, elle ferait correspondre un segment très court de A à un segment relativement long de B. Une restriction, appelée contrainte de pente, est donc nécessaire. Son rôle sera de définir la configuration possible que peuvent prendre des points consécutifs sur la fonction de déformation. Une illustration est faite sur les figures (III.2)

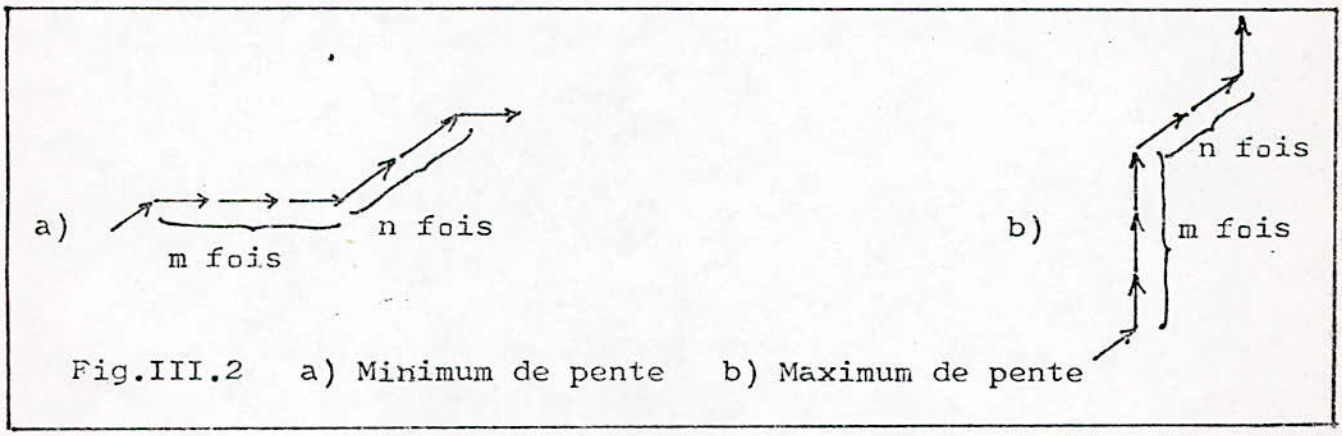


Fig.III.2 a) Minimum de pente b) Maximum de pente

Si un point c(k) se déplace dans la direction de l'axe i(ou j) m fois consécutives, il ne lui est pas permis de faire un pas supplémentaire dans la même direction avant d'en faire n dans la direction de la diagonale. La contrainte de pente peut être évaluée grâce à la grandeur suivante :

$$P = n/m \qquad \text{(III.9)}$$

Quand P=0, il n'y a pas de restriction sur la pente de la fonction F. Quand P=∞ (c-à-d m=o), la fonction est restreinte à la diagonale i=j.

Les figures III.3 donnent deux exemples de chemins permis pour c(k) sous la condition de penteP=1.

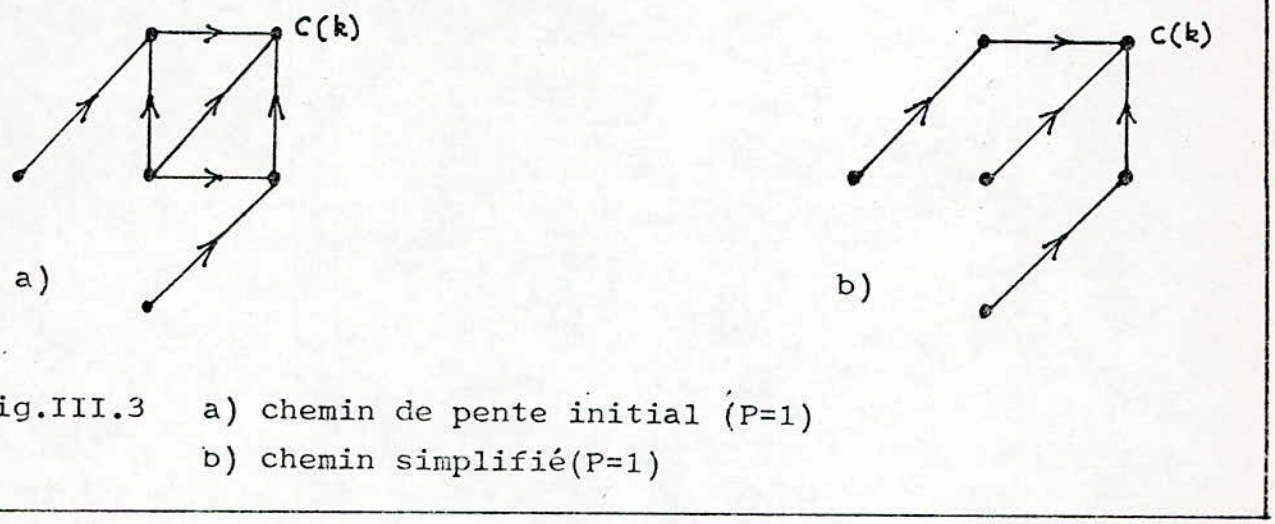


Fig.III.3 a) chemin de pente initial (P=1)
b) chemin simplifié(P=1)

Pour ce qui est de l'exemple de la figure a), il découle directement de la définition ci-dessus.
Pour celui de la figure b), une autre condition est imposée : la deuxième dérivée de la fonction est restreinte et le chemin ne doit pas être orthogonal. Cette nouvelle contrainte réduit le nombre de chemins à chercher.

III.2.3. Les coefficients de pondération

Si le dénominateur de l'équation (III.5) (appelé coefficient de pondération ou de normalisation), $N=\sum_{k=1}^{K} w(k)$, est indépendant de la fonc-tion de déformation F, nous pouvons le sortir des crochets. L'équation s'écrira alors :

$$D(A,B) = \frac{1}{N} \min_{F} \left[\sum_{k=1}^{K} d(c(k)).\ w(k) \right] \qquad \text{(III.10)}$$

Le problème ainsi simplifié peut être résolu par la programmation dynamique.

Il existe deux definitions typiques de coefficients de pondération qui permettent cette simplification. Ce sont :

1. La forme symétrique :

$$w(k) = (i(k) - i(k-1)) + (j(k) - j(k-1)) \qquad \text{(III.11)}$$

alors $N = I + J$ (III.12)

où I et J sont respectivement les longueurs des formes A et B.

2. La forme asymétrique

$$w(k) = (i(k) - i(k-1)) \qquad \text{(III.13)}$$

alors $N = I$ (III.14)

(ou bien $w(k) = (j(k) - j(k-1))$ alors $N = J$)

Dans la forme symétrique, on agit sur les deux axes pour trouver la meilleure superposition des deux formes de parole. Par contre, dans l'asymétrique, on agit sur l'un ou l'autre des deux axes.

Nous allons présenter dans ce qui suit, un algorithme de super-position par programmation dynamique pratique. Le principe de programmation dynamique peut être utilisé pour la résolution de l'équation (III.10) donnant la distance normalisée.

L'algorithme de base pour calculer cette équation s'écrit comme suit:

condition initiale :

$$g_1(c(1)) = d\,(c(1),\ w(1)) \qquad \text{(III.15)}$$

l'équation de programmation dynamique est :

$$g_k(c(k)) = \min_{c(k-1)} \left(g_{k-1}\,(\,c(k-1)) + d(c(k)),\ w(k)\right) \qquad \text{(III.16)}$$

La distance normalisée est alors :

$$D(A,B) = \frac{1}{N}\ g_K(c(K)) \qquad \text{(III.17)}$$

Il est implicitement admis que $c(0) = (0,0)$
Pour la forme symétrique $w(1) = 2$; pour la forme asymétrique $w(1)=1$.

En utilisant les restrictions sur la fonction de déformation décrites en III.2.2 et en substituant (III.11) ou (III.13) aux coefficients de pondération w(k) dans(III.16), plusieurs algorithmes pratiques peuvent être trouvés ./12/

IV. RECONNAISSANCE DU MOT CONNECTE

Nous avons vu que la programmation dynamique(appelée aussi DTW) était une méthode efficace pour manier les variations,dans le temps,des mots prononcés isolément.Cette technique a été appliquée avec succès pour la reconnaissance du mot connecté,ceci par des extensions de l'algorithme de SAKOE et CHIBA.

La technique de construction par niveaux(LB) est proposée par MYERS et RABINER/9/ dans le but d'aligner,dans le temps, et de façon optimale,une séquence de mot connecté à une séquence de formes de référence de mots isolés.

Dans ce qui suit,nous allons reprendre les algorithmes qui conduisent à l'algorithme complet LB-DTW,et discuter sur les aspects de sa mise en pratique,incluant la façon avec laquelle le BACKTRACKING (par marche arrière) est mis en exécution.

Tout d'abord on définit les divers symboles utilisés dans les algorithmes.

SYMBOLES	DEFINITIONS
$T(m)$	Forme de test
M	Longueur(en trames) de $T(m)$
$R_v(n)$	Forme de référence v
V	Nombre de mots d'une référence(Nbre de formes de référence)
N_v	Longueur(en trames) de la $v^{ème}$ forme de référence
R^s	Super forme de référence(concaténation de formes de référence)
L	Nombre de formes de référence dans une chaîne.
$\emptyset(l)$	Longueur(en trames) de l formes de réf concaténées (la trame de la super forme de réf qui correspond à la fin de la $l^{ème}$ forme de réf.)
w	Chemin de déformation dynamique(à ne pas confondre avec le coefficient de pondération de la partie III)
$d(m,n)$	Distance locale entre la $m^{ème}$ trame de la forme de test,et la $n^{ème}$ trame de la super forme de réf.
D	Distance globale entre la forme de test et la super forme de réf.
$D_A(m,n)$	Distance accumulée à la trame m de la forme de test et la trame n de la super forme de réf.
$L(m)$	Fonction limite inférieure
$U(m)$	Fonction limite supérieure

$\overline{L}(n)$	Contrainte inférieure sur la fonction de déformation
$\overline{U}(n)$	Contrainte supérieure sur la fonction de déformation
$D_1(m,n)$	Distance accumulée à la trame m du test et la trame n de la $1^{ème}$ réf de la super forme de réf
$\overline{D}_1(m)$	Distance accumulée à la trame m du test et la dernière trame de la $1^{ème}$ réf de la super forme de réf.
$d_1(m,n)$	Distance locale entre la $m^{ème}$ trame du test, et la $n^{ème}$ trame de la $1^{ème}$ réf de la super forme de réf
$L_1(m)$	Fonction limite inférieure modifiée pour le $1^{ème}$ niveau
$U_1(m)$	Fonction limite supérieure modifiée pour le $1^{ème}$ niveau
e_1	La trame du test à partir de laquelle le chemin optimal atteind $\emptyset(1)$ pour la $1^{ème}$ réf de la super forme de réf.
L_{MIN}	Nombre minimum de références dans une super forme de réf.
L_{MAX}	Nombre maximum de références dans une super forme de réf.
$D_{q(1)q(2)..q(1)}(m)$	Distance entre la forme de test et la super forme de réf $R_{q(1)} + R_{q(2)} + \ldots + R_{q(1)}$, à la trame m du test
$\overline{R}_L^S$	La meilleure super forme de réf de longueur L
$\overline{D}_1^B(m)$	Valeur minimum de $\overline{D}_1(m)$ pour toutes les superformes de réf, de longueur 1, possibles
$W_1(m)$	L'indice v de la forme de réf R_v donnant $\overline{D}_1^B(m)$
$F_1(m,n)$	Pointeur, en un point (m,n) du niveau 1, de la valeur initiale e_{1-1} du niveau 1-1 à partir de laquelle le meilleur chemin vers (m,n) est venu.
$\overline{F}_1(m)$	Position le long de la forme de test, à la trame m du niveau 1, à partir de laquelle le meilleur chemin vers le point (m,N_v) est venu.
$\overline{F}_1^B(m)$	Valeur de $\overline{F}_1(m)$ associée au mot de réf v qui donne $\overline{D}_1^B(m)$.
$D(b,e,v)$	Distance DTW entre R_v et la portion de la forme de test comprise entre les trames b et e.
$\hat{D}(b,e)$	La meilleure distance DTW entre n'importe quelle forme de réf et la forme de test comprise entre les trames b et e.
$\hat{V}(b,e)$	Forme de référence donnant la meilleure distance DTW pour la forme de test comprise entre b et e.

$c(m)$	Minimum local de $D_1(m-1,n)$ à la $(m-1)^{éme}$ trame du test.
$\hat{T}(m)$	Seuil de distance accumulée à la trame m du test.
T_{MIN}	Valeur minimale du seuil de distance
T_{MAX}	Pente de la courbe du seuil de distance.

IV.1 Application de DTW pour des chaînes de mot connecté

Soit une forme de test inconnue $T(m)$ $m = 1,2,.....,M$ où pour chaque valeur de m, $T(m)$ dénote un vecteur de caractéristiques et M est la longueur de T en trames(ou échantillons).Et supposons que T est enregistré avec une séquence de L formes de référence $R_{q(1)}(n),R_{q(2)}(n),.....,R_{q(L)}(n)$,où chaque $R_{q(k)}$ $(k = 1,2,....,L)$ fait partie d'un groupe de V formes de référence R_v $(v = 1,2,....,V)$,avec N_v comme longueur de la $v^{ème}$ forme de réf . Il est à noter que ces formes de référence ne sont sollicitées que pour l'algorithme 3.

L'idée de SAKOE est de définir une super forme de référence, $R^s_{q(1)q(2).....q(L)}$ (qui sera dénotée R^s) comme étant la concaténation de L formes de référence $R_{q(1)},...,R_{q(L)}$ .En d'autres termes :

$$R^s = R_{q(1)} \oplus R_{q(2)} \oplus R_{q(3)} ..... \oplus R_{q(L)} \qquad (1a)$$

ou

$$R^s(n)=\begin{cases} R_{q(1)}(n-\emptyset(0)) & 1+\emptyset(0)\leqslant n \leqslant \emptyset(1) \\ R_{q(2)}(n-\emptyset(1)) & 1+\emptyset(1)\leqslant n \leqslant \emptyset(2) \\ \vdots & \\ R_{q(1)}(n-\emptyset(1-1)) & 1+\emptyset(1-1)\leqslant n \leqslant \emptyset(1) \\ \vdots & \\ R_{q(L)}(n-\emptyset(L-1)) & 1+\emptyset(L-1)\leqslant n \leqslant \emptyset(L) \end{cases} \qquad (1b)$$

où la fonction de longueur $\emptyset(1)$ est définie comme suit :

$$\emptyset(1) = \sum_{k=1}^{\ell} N_{q(k)} \qquad (2a)$$

$$\emptyset(0) = 0 \qquad (2b)$$

Ainsi l'enregistrement temporel de la forme detest T,avec une super forme de référence R^s peut se poser comme problème d'alignement de DTW (fig.1).

Si w représente le chemin de l'alignement temporel optimum,

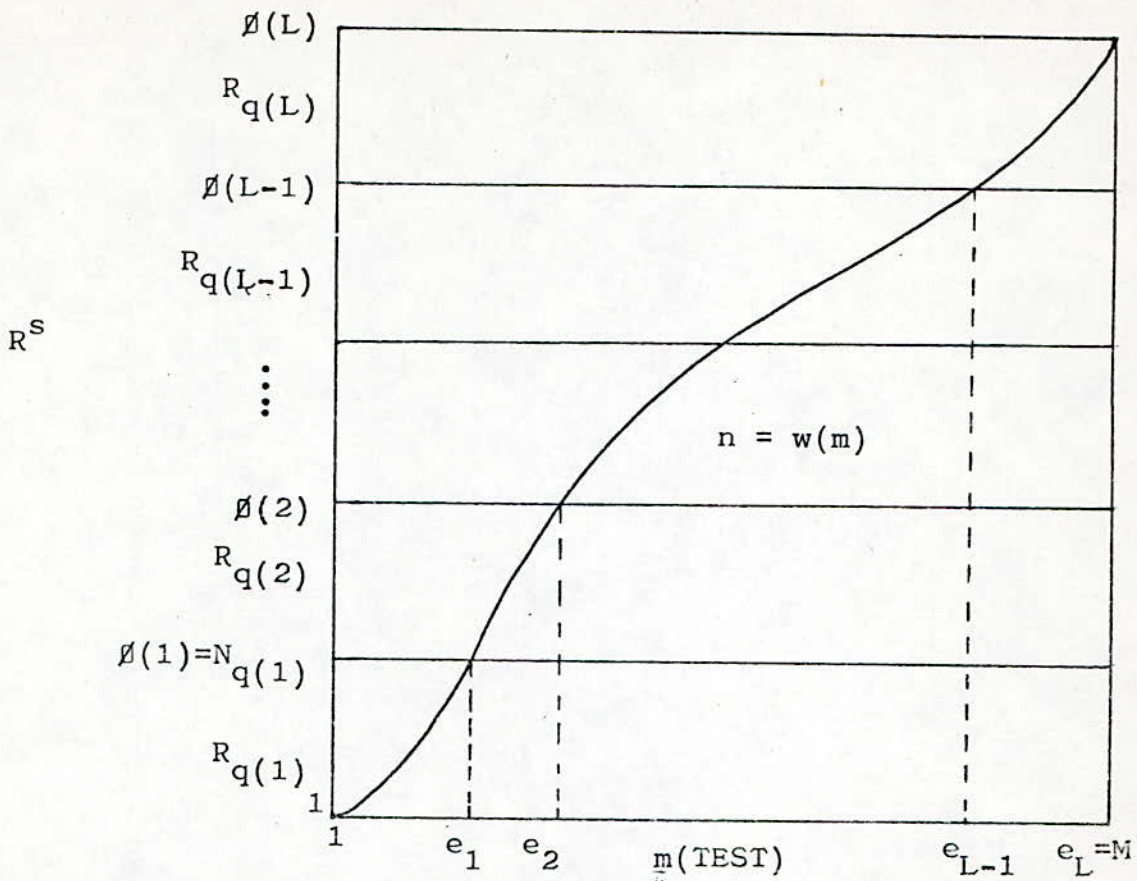


Fig.1 Illustration de l'alignement DTW entre T et R^s

alors n = w(m) est un cadrage fonctionnel entre la trame test m et la trame n de la super référence.Quand q(1) est connu pour tout l, R^s peut être traitée comme une seule forme de référence,l'alignement temporel entre T(m) et R^s(n)(c-à-d détérmination de w)peut alors se faire utilisant une seule application d'un algorithme de DTW avec contrainte sur les bornes.Ainsi si d(m,n) représente la distance locale entre les trames m et n,alors DTW,avec contrainte sur les bornes, minimise la distance globale D définie par :

$$D = \min_{w(m)} \left[\sum_{m=1}^{M} d(m, w(m)) \right] \qquad (3)$$

avec les contraintes sur les bornes:

$$\begin{cases} w(1) = 1 \\ w(M) = Ø(L) \end{cases} \qquad (4)$$

et celles du chemin local,prescrites sur w(m).

Pour le moment,nous admettons que w(m) est une fonction non décroissante.L'algorithme minimisant (3) est alors :

Algorithme 1. Algorithme DTW avec contrainte aux bornes.

1) Initialiser $D_A(0,0)=0$, $D_A(0,n) = \infty$, $n \neq 0$.

2) En utilisant la récurrence, pour $m = 1,2,\ldots M$ et pour $n = L(m),\ldots,U(m)$,calculer

$$\hat{n} = \underset{\bar{L}(n)\leq n'\leq \bar{U}(n)}{\operatorname{argmin}} (D_A(m-1,n'))$$

$$D_A(m,n) = d(m,n) + D_A(m-1,\hat{n})$$

3) Solution finale

$$D = D_A(M,\emptyset(L))$$

$\underset{x}{\operatorname{argmin}} f(x)$ est la valeur de x qui minimise f(x).

Les spécifications sur L(m),U(m),$\bar{L}(n)$ et $\bar{U}(n)$ ne sont pas nécessaires à ce niveau, pour comprendre l'implémentation de DTW.

IV.2 Une approche de Level Building pour DTW

L'algorithme 1 comme décrit précédemment,peut être implémenté en niveaux,c-à-d une référence(de la super forme de référence) à chaque fois.Pour ce faire nous devons utiliser d'autres distances définies comme suit :

$$d_1(m,n) = d(m,n + \emptyset(l-1))$$

$$D_1(m,n) = D_A(m,n+\emptyset(l-1)) \qquad (5)$$

$$\bar{D}_1(m) = D_1(m,N_{q(1)}) = D_A(m,\emptyset(1)) \qquad (6a)$$

$$\bar{D}_0(m) = \begin{cases} 0 \text{ pour } m = 0 \\ \infty \text{ pour } m \neq 0 \end{cases} \qquad (6b)$$

l'algorithme 2 nous indique alors,comment calculer le chemin optimal dans les niveaux.

Algorithme 2. DTW avec contrainte sur les bornes, utilisant les solutions par niveaux.

1) Pour $l = 1,2,\ldots,L$ faire les étapes de 2)à4)

2) Initialiser $D_l(m,0) = \overline{D}_{l-1}(m)$; $m = 0,1,\ldots,M$

3) En utilisant la récurrence, pour $m = 1,2,\ldots,M$ et pour $n = L_l(m),\ldots,U_l(m)$, calculer

$$\hat{n} = \underset{\overline{L}(n) \leq n' \leq \overline{U}(n)}{\operatorname{argmin}} \; (D_l(m-1,n'))$$

$$D_l(m,n) = D_l(m-1,\hat{n}) + d_l(m,n)$$

4) $\overline{D}_l(m) = D_l(m,N_{q(l)})$, $m = 1,2,\ldots,M$.

5) $D = \overline{D}_L(M)$.

les fonctions limites inférieure et supérieure relatives au niveau l sont données par :

$$L_l(m) = \max\,(1, L(m) - \emptyset(l-1)) \qquad (7a)$$

$$U_l(m) = \min\,(N_{q(l)}, U(m) - \emptyset(l-1)) \qquad (7b)$$

La figure 2 illustre la différence qui existe entre les algorithmes 1 et 2

Remarque: Le champ de recherche de la fonction de déformation (parallélogramme hachuré sur cette figure) est limité par des contraintes qui éliminent des zones très improbables dans le domaine de recherche et dans lesquelles toute comparaison n'a aucun sens.

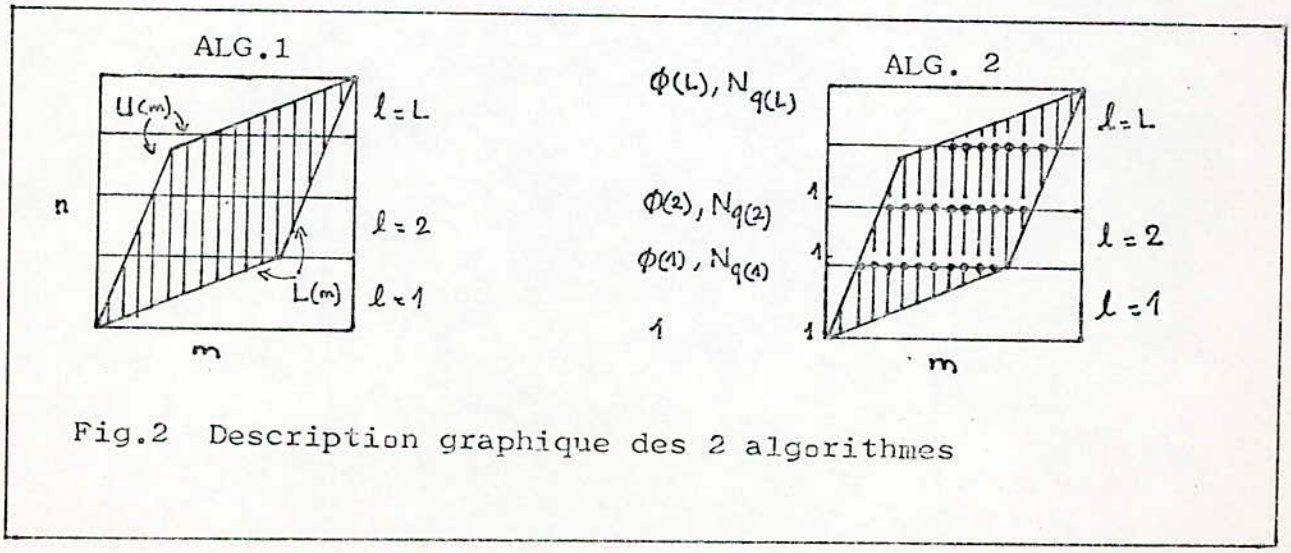


Fig.2 Description graphique des 2 algorithmes

Pour l'algorithme 1 ,le calcul se fait suivant des bandes ver--ticales parcourant généralement plus d'une référence,alors que pour l'algorithme 2,il se fait suivant une séquence de bandes , toujours verticales,mais pour chaque référence.Pour la $l^{ème}$ référence, l'en--semble des distances accumulées le long de la ligne horizontale $n = \emptyset(l)$ est alors sauvegardé afin d'être utilisé comme un ensemble de distances initiales pour le niveau suivant.

IV.3 Utilisation des formes de référence multiples

Nous venons de voir comment une super forme de référence, R^s, pouvait être alignée dans le temps avec une forme de test T, utilisant un algorithme de LB-DTW. Voyons maintenant comment cette mêmeapproche peut être appliquée à un groupe de formes de référence variables , c-à-d quand chaque référence de la super forme de référence ap--partient à un groupe de V formes de référence.Cependant quelques points doivent,d'abord,être soulignés.

Premièrement,on définit un groupe de frontières sur le test, e_l,tel que $w(e_l) = \emptyset(l)$ c-à-d e_l est la valeur de m pour laquelle le chemin optimal w fait correspondre e_l à $\emptyset(l)$ qui est la fin de la $l^{ème}$ forme de référence(fig.1).Les valeurs e_l ne sont connues que si le chemin de déformation est connu .

$$e_l = w^{-1}(\emptyset(l)) \qquad l = 1,2,\ldots,L. \tag{8}$$

D'autre part,pour comparer les résultats des distances d'un groupe de super formes de référence à un test donné,les valeurs de D doivent être normalisées.Cependant,à partir de (3),on voit que le facteur de normalisation est M,la longueur du test,ceci indépendem--ment de la super forme de référence.Il n'est donc pas nécessaire de normaliser les valeurs de D pour deux super formes de référence dif--férentes,pour comparer leurs distances.

Un autre point à savoir est qu'en général la longueur de la super forme de référence,L,n'est pas connue.Seules les limites sur L sont connues:

$$L_{MIN} \leqslant L \leqslant L_{MAX} \tag{9}$$

ce qui pose le problème de variabilité de la longueur des super formes de référence.

Finalement pour une super forme de référence donnée $R^s_{q(1)q(2)\ldots q(L)}$,on définit la distance $D_{q(1)q(2)\ldots q(L)}(m)$ comme étant la distance fournie par l'algorithme 2 en faisant correspondre la super référence entière R^s à la partie de la forme test comprise entre la trame 1 et la trame m.Ainsi $D = D_{q(1)q(2)\ldots q(L)}(M)$. La recon--naissance du mot connecté est donc la solution de la minimisation:

$$\min_{L_{MIN} \leqslant L \leqslant L_{MAX}} \left[\min_{q(1),q(2),\ldots,q(L)} \left(D_{q(1)q(2)\ldots q(L)}(M) \right) \right] \qquad (10)$$

autrement dit, il suffit de minimiser la distance D sur toutes les super formes de référence possibles de toutes les longueurs possibles . La séquence $q(1),\ldots,q(L)$ qui minimise (10) est choisie comme étant la forme de test estimée. Pour la résolution efficace de la minimisation ci-dessus, il est souhaitable d'utiliser une approche LB du type décrit dans l'algorithme 2 : La décision sur la 1 ème forme de référence doit précéder celle de la $(1+1)^{ème}$.

Afin de montrer comment cela est réalisable, le théorème suivant est nécessaire.

Théorème : S'il existe une super forme de référence $R^s_{r1\ r2\ \ldots\ r1}$, tel que: $D_{r1r2\ldots r1}(e_1) \leqslant D_{s1\ s2\ldots s1}(e_1)$ pour n'importe quel $R^s_{s1s2\ldots s1}$ alors

$D_{r1\ldots r1r1+1}(e_{1+1}) \leqslant D_{s1s2\ldots s1r1+1}(e_{1+1})$ pour n'importe quelle $R^s_{s1s2\ldots s1}$ et R_{r1+1} .

Démonstration du théorème : Admettons que $R^s_{s1\ldots s1}$ est une super forme de référence générale de longueur 1 et que $R_{r1\ldots r1}$ est la super forme de référence qui minimise la distance $D_{s1\ldots s1}(e_1)$ pour tous les s1s2...s1 possibles . On a par définition :

$$D_{s1s2..s1}(e_1) = \sum_{m=1}^{e_l} d(m,w(m)) \qquad (A1)$$

$$\text{et } D_{s1s2..s1r1+1}(e_{1+1}) = \sum_{m=1}^{e_{l+1}} d(m,w(m)) \qquad (A2)$$

$$= \sum_{m=1}^{e_l} d(m,w(m)) + \sum_{m=e_{l+1}}^{e_{l+1}} d(m,w(m)) \qquad (A3)$$

$$= D_{s1s2..s1}(e_1) + \sum_{m=e_{l+1}}^{e_{l+1}} d(m,w(m)) \ . \qquad (A4)$$

puisque le $2^{ème}$ terme de (A4) ne dépend que de R_{r1+1} et puisque la séquence $R_{r1} \oplus R_{r2} \oplus R_{r3} \oplus \ldots \oplus R_{r1}$ minimise $D_{s1s2..s1}(e_1)$, ces 1 formes de référence vont être les 1 premières formes de référence de la meilleure séquence de longueur 1+1, où la $(1+1)^{ème}$ forme de référence correspond à la portion de la forme de test comprise entre les trames e_1+1 et e_{1+1} .

En utilisant ce théorème, on peut maintenant trouver la chaîne optimale de longueur L , comme suit :

Algorithme 3.

1) Pour $l = 1,2,\ldots,L_{MAX}$ faire les étapes de 2)et 3)

2) Pour $v = 1,2,\ldots,V$ calculer

$D_{r1r2\ldots r(l-1)v}(e_l)$ à partir des étapes 2) à 4) de l'algorithme 2.

3) $r1 = \underset{1 \leqslant v \leqslant V}{\operatorname{argmin}} \left(D_{r1r2..r(l-1)v}(e_l)\right)$

4) $\bar{R}_L^s = R^s_{r1r2\ldots rL}$ (la meilleure super forme de référence de longueur L).

A chaque niveau l,cet algorithme trouve la forme de référence qui minimise la distance à ce niveau.Cependant,le défaut majeur de cet algorithme est qu'il admet que les frontières $e_1,e_2,\ldots,e_L$ soient connues à chaque niveau,ce qui n'est pas le cas puisque les valeurs e_l ne sont déterminées qu'après avoir connu le chemin entièrement. Dè là,une modification doit être apportée de telle sorte que pour tous les groupes de $e_1,e_2,\ldots,e_L$ possibles,la meilleure séquence de L formes de référence est déterminée.

Pour ce faire,on utilise $\bar{D}_l^B(m)$ comme étant la distance accumulée à la m^{eme} trame du test et à la fin de la l^{eme} forme de référence de la meilleure super forme de référence partielle(à la trame m).

$$\bar{D}_l^B(m) = \min_{q(1)..q(l)} \left(D_{q(1)q(2)\ldots q(l)}(m)\right) \qquad (11a)$$

avec comme valeurs initiales

$$\bar{D}_o^B(m) = \begin{cases} 0 \text{ pour } m = 0 \\ \infty \text{ pour } m \neq 0 \end{cases} \qquad (11b)$$

L'algorithme modifié est alors :

Algorithme 4.

1) Initialiser $\bar{D}_o^B(0) = 0$, $\bar{D}_o^B(m) = \infty$, $m = 1,2,\ldots,M$

$\bar{D}_l^B(0) = \infty$, $l = 1,2,\ldots,L_{MAX}$

2) Pour $l = 1,2,\ldots,L_{MAX}$ faire les étapes de 3) à 7)

3) Pour v = 1,2,...,V faire les étapes de 4) à 6)

4) Initialiser $D_1(m,0) = \bar{D}^B_{l-1}(m)$ pour m = 0,1,...,M

5) Calculer en utilisant $R_{q(1)}$ (au $1^{ème}$ niveau) pour m =1,2,..,M n =$L_1(m),...,U_1(m)$.

$$\hat{n} = \underset{\bar{L}(n) \leqslant n' \leqslant \bar{U}(n)}{\text{argmin}} (D_1(m-1,n'))$$

$$D_1(m,n) = d_1(m,n) + D_1(m-1,\hat{n})$$

6) $\bar{D}^v_1(m) = D_1(m,N_v)$

7) $\bar{D}^B_1(m) = \min_{1 \leqslant v \leqslant V} (\bar{D}^v_1(m))$, m = 1,2,...,M.

8) $W_1(m) = \underset{1 \leqslant v \leqslant V}{\text{argmin}} (\bar{D}^v_1(m))$, m = 1,2,...,M.

9) $L = \underset{L_{MIN} \leqslant L \leqslant L_{MAX}}{\text{argmin}} (\bar{D}^B_1(M))$

$D = \bar{D}^B_L(M)$

10) $\bar{R}^S = R_{W_1(e_1)} \oplus R_{W_2(e_2)} \oplus \ldots \oplus R_{W_L(e_L)}$.

Le réseau $W_1(m)$ indique l'indice v de la forme de référence R_v qui donne le meilleur chemin à la fin du $1^{ème}$ niveau et à la $m^{ème}$ trame du test.Le réseau $\bar{D}^v_1(m)$ est la distance accumulée à la fin du $1^{ème}$ niveau utilisant la forme de référence R_v à la trame m du test .

L'algorithme 4 construit tous les chemins possibles, une forme de référence à la fois.En effet :
L'étape 2) initie les chemins de construction par niveaux,1.

L'étape 3) boucle à travers toutes les formes de référence au niveau 1

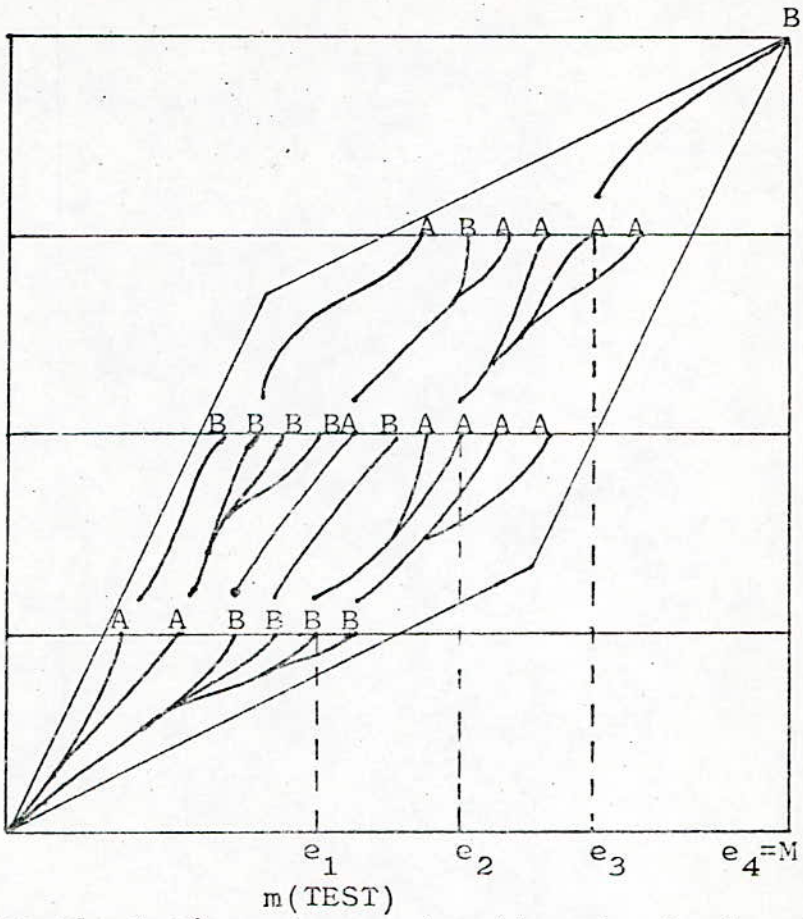
Les étapes 4),5) et 6) rendent performante la récurrence de program--mation dynamique(DP) pour une forme de référence fixée q(1) au niveau 1.(Ceci est similaire aux calculs de LB de l'algorithme 2).

Les étapes 7) et 8) trouvent le meilleur chemin pour toutes les références possibles jusqu'à la fin du $1^{ème}$ niveau et enregistrent les formes de référence R_v associées au meilleur chemin.

L'étape 9) choisit la meilleure longueur pour la meilleure super forme de référence.

Et l'étape 10) reprend la meilleure super forme de référence .

La figure 3 illustre un exemple typique de l'utilisation de l'algorithme 4.On admet que L est connu (L = 4),que w(m) est astreint à être dans le parallèlogramme et que V = 2 , c-à-d qu'il existe seu--lement deux formes de référence qu'on dénote par A et B .



F.3. Un exemple typique de construction de chemins de déformation pour 2 formes de référence.

Cette figure montre à chaque niveau la meilleure forme de référence à la $m^{ème}$ trame, $R_{W_1}(m)$, et le meilleur chemin à partir du niveau précé--dent. par exemple, au premier niveau, la forme de référence peut finir à n' importe laquelle des 6 différentes trames le long du test.
Si elle finit à l'une des deux premières trames, alors la meilleure référence est A et si elle finit à n'importe laquelle des quatre autres positions, la meilleure référence est B.
A chaque niveau, pour toutes les trames de fins possibles, la meilleure référence est retenue. Jusqu'à ce qu' au niveau 4, une unique meilleure référence est obtenue.
Pour former la meilleure super forme de référence, on concatène les références obtenues en faisant " marche arrière " (backtracking) à partir de la fin du test ($R^S = R_B \oplus R_A \oplus R_A \oplus R_B$). Les frontières

e_1, e_2, e_3 et $e_4 = M$ le long du test sont alors détérminées.

IV.4 Implémentation du backtracking

La dernière étape de l'algorithme 4 reprend la super chaîne optimale de référence, $\overline{R}^s$, en concaténant les formes à partir d'un groupe de fontières de test e_l, $l = 1,2,\ldots,L$. Il n'y a qu'une seule frontière e_{l-1} associée à chaque point (m,n) du niveau l. Ceci est illustré dans la figure 4. Il est admis que le chemin optimal vers le point (m,n) au $l^{ème}$ niveau est venu du point $(e_{l-1}, \emptyset(l-1))$ à la fin du (l-1) ème niveau.

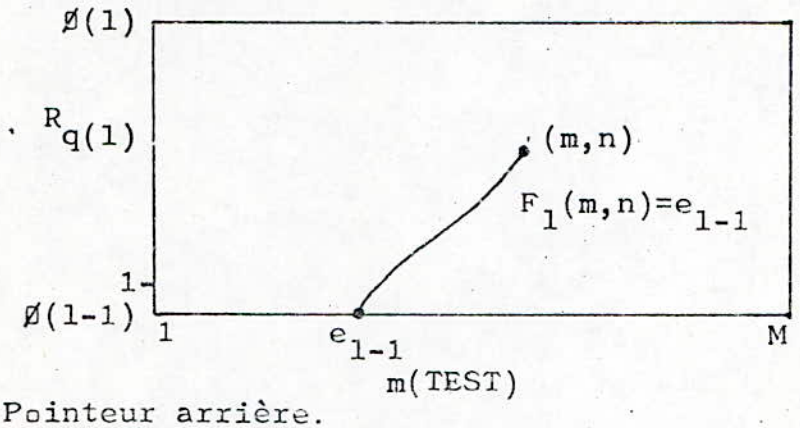


Fig.4. Pointeur arrière.

De là, on définit un réseau de " from frame " $F_l(m,n)$, comme étant la valeur de e_{l-1} à partir de laquelle le meilleur chemin vers (m,n) au niveau l utilisant la référence q(l), est obtenu.

Le backtracking peut être alors représenté par l'algorithme suivant :

Algorithme de backtracking

1) Pour l =1,2,...,L faire les étapes de 2) à 4)

2) Pour m = 0,1,...,M initialiser $F_l(m,0) = m$

3) Pour m = 1,2,...,M et $n = L_l(m), \ldots, U_l(m)$, calculer $\hat{n}$ et $D_l(m,n)$ comme à l'étape 3) de l'algorithme 2, et calculer $F_l(m,n) = F_l(m-1,\hat{n})$

4) $\overline{F}_l(m) = F_l(m,N_v)$, $m = 1,2,\ldots,M$

5) $e_L = M$

6) $e_l = \overline{F}_{l+1}(e_{l+1})$, $l = L-1, L-2, \ldots, 1$.

$\overline{F}_1(m)$ enregistre la position, le long du test, à la fin du$(l-1)^{ème}$ niveau d'où le meilleur chemin vers (m,N_v) est venu.

$F_1(m,n)$ est calculé comme $F_1(m-1,\hat{n})$ car le meilleur chemin vers (m,n) est venu du même endroit que le meilleur chemin vers chaque point $(m-1,\hat{n})$, intermédiaire.
La figure 5 montre un exemple d'alignement temporel avec plusieurs chemins locaux. On peut voir que $\overline{F}_L(e_L) = e_{L-1}$ et qu'en utilisant $\overline{F}_1(m)$ il est possible d'avoir les frontières à chaque niveau.

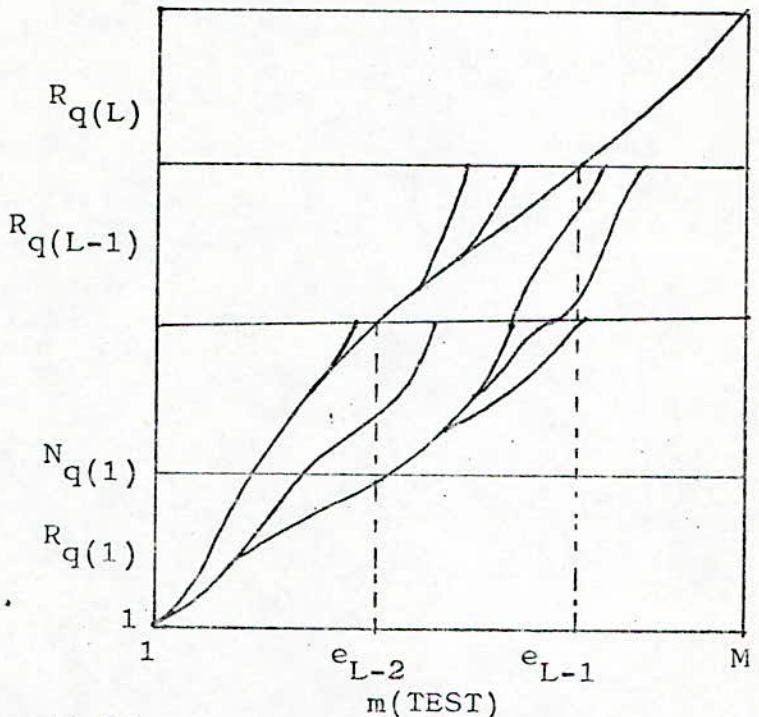


Fig.5. Exemple d'alignement temporel avec plusieurs chemins locaux.

Il est donc permis d'appliquer le backtracking à l'algorithme 4 (c-à-d quand les super formes de référence sont variables), avec cependant les définitions suivantes :

$\overline{F}_1^B(m)$ est la valeur de $\overline{F}_1(m)$ associée au meilleur mot référence $W_1(m)$ (la plus petite distance) au $1^{ème}$ niveau et à la position m du test. on définit aussi $\overline{F}_1^V(m)$ comme étant la valeur de $\overline{F}_1(m)$ pour la référence R_v. C'est ainsi qu'on a l'algorithme 5.

Algorithme 5. L'algorithme LB-DTW

1) Initialiser $\bar{D}_0^B(m) = 0$ pour $m = 0$, $\bar{D}_0^B(m) = \infty$ pour $m = 1,2,\ldots,M$

et $\bar{D}_1^B(0) = \infty$, $1 = 1,2,\ldots,L_{MAX}$:

2) Pour $1 = 1,2,\ldots,L_{MAX}$ faire les étapes de 3) à 7)

3) Pour $v = 1,2,\ldots,V$ faire les étapes de 4) à 6)

4) Pour $m = 0,1,\ldots,M$ initialiser $D_1(m,0) = \bar{D}_{1-1}^B(m)$,

$F_1(m,0) = m$.

5) Pour $m = 1,2,\ldots,M$ et $n = L_1(m),\ldots,U_1(m)$ calculer en utilisant $R = R_v$:

$$\hat{n} = \underset{\bar{L}(n) \leqslant n' \leqslant \bar{U}(n)}{\operatorname{argmin}} (D_1(m-1,n'))$$

$$D_1(m,n) = d_1(m,n) + D_1(m-1,\hat{n})$$

$$F_1(m,n) = F_1(m-1,\hat{n}).$$

6) Pour $m = 1,2,\ldots,M$ calculer :

$$\bar{D}_1^v(m) = D_1(m,N_v)$$

$$\bar{F}_1^v(m) = F_1(m,N_v)$$

7) Pour $m = 1,2,\ldots,M$ calculer:

$$\bar{D}_1^B(m) = \min_{1 \leqslant v \leqslant V} (\bar{D}_1^v(m))$$

$$W_1(m) = \underset{1 \leqslant v \leqslant V}{\operatorname{argmin}} (\bar{D}_1^v(m))$$

$$\bar{F}_1^B(m) = \bar{F}_1^{W_1(m)}(m)$$

8) $$L = \underset{L_{MIN} \leqslant 1 \leqslant L_{MAX}}{\operatorname{argmin}} (\bar{D}_1^B(M))$$

$$D = \bar{D}_L^B(M)$$

9) $e_L = M$

10) $e_l = \bar{F}^B_{l+1}(e_{l+1})$, $l = L-1, L-2, \ldots, 1$.

11) $\bar{R}^S = R_{W_1}(e_1) \oplus R_{W_2}(e_2) \oplus \ldots \oplus R_{W_L}(e_L)$.

L'algorithme 5 représente une approche complète de LB dans le but d'adapter, de façon optimale, des formes de référence (variables) à une forme de test donnée. Il nous donne le nombre optimal de formes de référence, L, la distance minimale D de la chaîne, les références dans la chaîne $R_{W_1}(e1)$, $R_{W_2}(e2)$, ..., $R_{W_L}(eL)$ et le groupe de points frontières, e_l, qui adaptent la dernière trame des formes de référence dans la chaîne.

IV.5 Relation entre l'algorithme 5 et le Two-Level DTW de SAKOE.

SAKOE avait proposé un autre algorithme de DTW, appelé algorithme de DP à deux niveaux, afin d'obtenir le meilleur assortiment pour une chaîne de mots test donnée. Cet algorithme peut se résumer comme suit :

Algorithme S1 (SAKOE)

S1.1. Superposition au niveau "mot"

1) calculer les distances de mots D(b,e,v) pour $1 \leqslant b \leqslant e \leqslant M$ et pour $1 \leqslant v \leqslant V$ où D(b,e,v) est la distance DTW entre la forme de référence R_v et la forme de test, sujette aux contraintes limites $w(b)=1$, $w(e)=N_v$, afin que la référence R_v épouse la portion de la forme de test comprise entre les trames b et e.

2) calculer les meilleures distances début-fin ($\hat{D}(b,e)$)

$$\hat{D}(b,e) = \min_{1 \leqslant v \leqslant V} (D(b,e,v)) \qquad (12a)$$

$$\hat{V}(b,e) = \operatorname{argmin}_{1 \leqslant v \leqslant V} (D(b,e,v)) \qquad (12b)$$

et où $\hat{V}(b,e)$ enregistre la meilleure forme de référence s'adaptant au test entre les trames b et e.

S1.2. Superposition au niveau "phrase"

1) Initialiser $\bar{D}_o^B(0) = 0$

2) Calculer, pour $e = 1,2,\ldots,M$, et pour $l = 1,2,\ldots,L_{MAX}$, les quantités:

$$\bar{D}_l^B(e) = \min_{1 \leqslant b \leqslant e} (\hat{D}(b,e) + \bar{D}_{l-1}^B(b-1)) \qquad (13a)$$

$$\bar{F}_l^B(e) = \operatorname{argmin}_{1 \leqslant b \leqslant e} (\hat{D}(b,e) + \bar{D}_{l-1}^B(b-1)) - 1 \qquad (13b)$$

où $\bar{F}_l^B(e)$ est le point, le long du test, indiquant la fin de la $(l-1)^{ème}$ forme de référence dans la séquence qui génère la distance minimum $\bar{D}_l^B(e)$.

3) Trouver la meilleure longueur de la séquence(L)

$$L = \operatorname{argmin}_{L_{MIN} \leqslant l \leqslant L_{MAX}} (\bar{D}_l^B(M)).$$

4) Poser les points limites (les bornes) de la chaîne

$$e_L = M$$
$$e_o = 0$$

5) Procéder au backtracking pour $l = L-1, L-2, \ldots, 1$ pour avoir

$$e_l = \bar{F}_{l+1}^B(e_{l+1}).$$

6) Retrouver les indices des meilleures formes de référence $v_1, v_2, \ldots, v_L$ tel que:

$$v_l = \hat{V}(e_{l-1}+1, e_l) \quad l = 1,2,\ldots,L.$$

7) Former la meilleure chaîne

$$\bar{R}_L^S = R_{v_1} \oplus R_{v_2} \oplus \ldots \oplus R_{v_L}.$$

On va voir maintenant comment l'algorithme 5 (l'algorithme de LB) peut être exprimé comme une seule mais importante modification de l'algorithme de SAKOE.

En utilisant (12a) dans (13a) on a :

$$\bar{D}_l^B(e) = \min_{1 \leqslant b \leqslant e} \left(\min_{1 \leqslant v \leqslant V} (D(b,e,v)) + \bar{D}_{l-1}^B(b-1) \right). \qquad (14)$$

Si on inverse l'ordre des minimisations on aura :

$$\bar{D}_1^B(e) = \min_{1 \leq v \leq V} \left(\min_{1 \leq b \leq e} (D(b,e,v) + \bar{D}_{1-1}^B(b-1)) \right). \quad (15)$$

Avec cet ordre inversé , on peut réecrire l'algorithme S1 dans une forme modifiée,comme suit :

Algorithme S2

S2.1. Superposition au niveau "mot"

1) Pour $1 \leq b \leq e \leq M$ et pour $1 \leq v \leq V$, calculer les distances de mots $D(b,e,v)$.

S2.2. Superposition au niveau "phrase"

1) Initialiser $\bar{D}_o^B(0) = 0$

2) Pour $1 = 1,2,\ldots,L_{MAX}$ faire les étapes 3) et 4)

3) Pour $e = 2,3,\ldots,M$ et pour $v = 1,2,\ldots,V$, calculer la première minimisation

$$\bar{D}_1^v(e) = \min_{1 \leq b \leq e} (D(b,e,v) + \bar{D}_{1-1}^B(b-1)) \quad (16a)$$

$$\bar{F}_1^v(e) = \operatorname{argmin}_{1 \leq b \leq e} (D(b,e,v) + \bar{D}_{1-1}^B(b-1)) - 1 \quad (16b)$$

où $\bar{D}_1^v(e)$ est la meilleure distance,à la trame e du test,quand la forme de la lème référence est R_v et $\bar{F}_1^v(e)$ est la trame à laquelle la forme de la (1-1)ème référence finit.

4) Calculer la seconde minimisation

$$\bar{D}_1^B(e) = \min_{1 \leq v \leq V} (\bar{D}_1^v(e))$$

$$\bar{W}_1(e) = \operatorname{argmin}_{1 \leq v \leq V} (\bar{D}_1^v(e))$$

$$\bar{F}_1^B(e) = \bar{F}_1^{W_1(e)}(e)$$

où $W_1(e)$ est la forme de référence,au niveau 1 et à la trame test e, qui minimise $\bar{D}_1^B(e)$.

5) Trouver la meilleure longueur L

$$L = \underset{L_{MIN} \leqslant l \leqslant L_{MAX}}{\text{argmin}} \; (\bar{D}_l^B(M))$$

6) Points limites

$$e_L = M$$

7) Backtracking pour $l = L-1, L-2, \ldots, 1$. et on a

$$e_l = \bar{F}_{l+1}^B(e_{l+1})$$

8) Les meilleures formes de référence $v_l = W_l(e_l)$, $l = 1,2,..,L$

9) La meilleure chaîne de référence

$$R_L^s = R_{v_1} \oplus R_{v_2} \oplus \ldots \oplus R_{v_L} .$$

On peut remarquer que le calcul de $\bar{D}_l^v(e)$ peut être fait pour tous les e (c-à-d $1 \leqslant e \leqslant M$), pour v fixé et $\bar{D}_{l-1}^B(m)$ donné $m=0,1,..,M$ sans pour cela avoir à calculer D(b,e,v), séparément. On peut le voir sur l'algorithme S3.

Algorithme S3 (Remplacement des étapes 1) de S2.1 et 2) et 3) de S2.2)

1) Initialiser, pour $m = 0,1,\ldots,M$, $D_l(m,0) = \bar{D}_{l-1}^B(m)$

$$F(m,0) = m$$

2) Pour $m = 1,2,\ldots,M$ et $n = L(m),\ldots,U(m)$, calculer par récurrence

$$\hat{n} = \underset{\bar{L}(n) \leqslant n' \leqslant \bar{U}(n)}{\text{argmin}} \; (D_l(m-1,n'))$$

$$D_l(m,n) = D_l(m-1,\hat{n}) + d_l(m,n)$$

$$F_l(m,n) = F_l(m-1,\hat{n}).$$

3) Pour $e = 2,3,\ldots,M$

$$\bar{D}_l^v(e) = D_l(e,N_v)$$

$$\bar{F}_l^v(e) = F_l(e,N_v)$$

Il est facile de constater que l'algorithme S3 avec les étapes de 2) à 9) (de l'algorithme S2) est identique à l'algorithme 5, pré-cédemment vu (IV.4).Cela confirme bien sûr le fait que l'algorithme de SAKOE est essentiellement identique à l'algorithme de LB.Cependant, ce dernier est plus efficace et c'est la raison pour laquelle il a été décrit de manière plus détaillée.Une"petite" étude comparative, complémentaire,concernant ces deux algorithmes et basée(pour un exmple donné) sur le nombre d'opérations et le stockage fera l'objet de la partie IV.8.Mais pour le moment,on parle des aspects de l'implé-mentation de l'algorithme de LB.

IV.6 Aspects d'implémentation et modifications de l'algorithme LB-DTW

Nous avons vu que l'approche de LB-DTW peut être utilisée pour adapter des séries concaténées de formes de référence à une chaîne de test connectée. Les principes de base utilisés dans la formulation de l'algorithme étaient le rétablissement du chemin de backtracking et l'optimalité locale du chemin (le meilleur chemin de (1,1) à (m,n) qui passe par (m',n') inclue, comme sous-section, le meilleur chemin de (1,1) à (m',n')). De là , en procédant de gauche à droite, l'algorithme trouve la meilleure chaîne partielle, à chaque niveau, pour chaque valeur de m, et recherche le chemin correct par le backtracking à partir de la fin de la chaîne.

Il y a un certain nombre d'aspects de l'algorithme 5 qui doivent être passés en revue afin de mieux comprendre comment cet algorithme est utilisé en pratique.

IV.6.1 Intervalle de restriction de n

Si on examine le calcul de l'algorithme 5 pour obtenir la distance accumulée au point (m,n), au niveau 1,$D_1(m,n)$, on voit que l'intervalle global de n est restreint par un groupe de contraintes aux bornes inférieure et supérieure,L(m) et U(m). Typiquement ces contraintes aux limites restreignent la zone du plan (m,n) dans laquelle le meilleur chemin peut se trouver. Elles peuvent être de la forme suivante :

$$L(m) = \frac{m+1}{2} \qquad (17a)$$

$$U(m) = 2(m-1)+1 \qquad (17b)$$

Elles sont indépendantes de n et astreignent le chemin w à être dans une région définie (délimitée) par les lignes de pente 1/2 et 2, dont l'origine est (1,1). La figure 6 illustre comment ces contraintes influent sur le calcul de l'algorithme LB.

On voit qu'au premier niveau, la récurence de DTW a besoin d'être calculée seulement dans la région hachurée G_1.

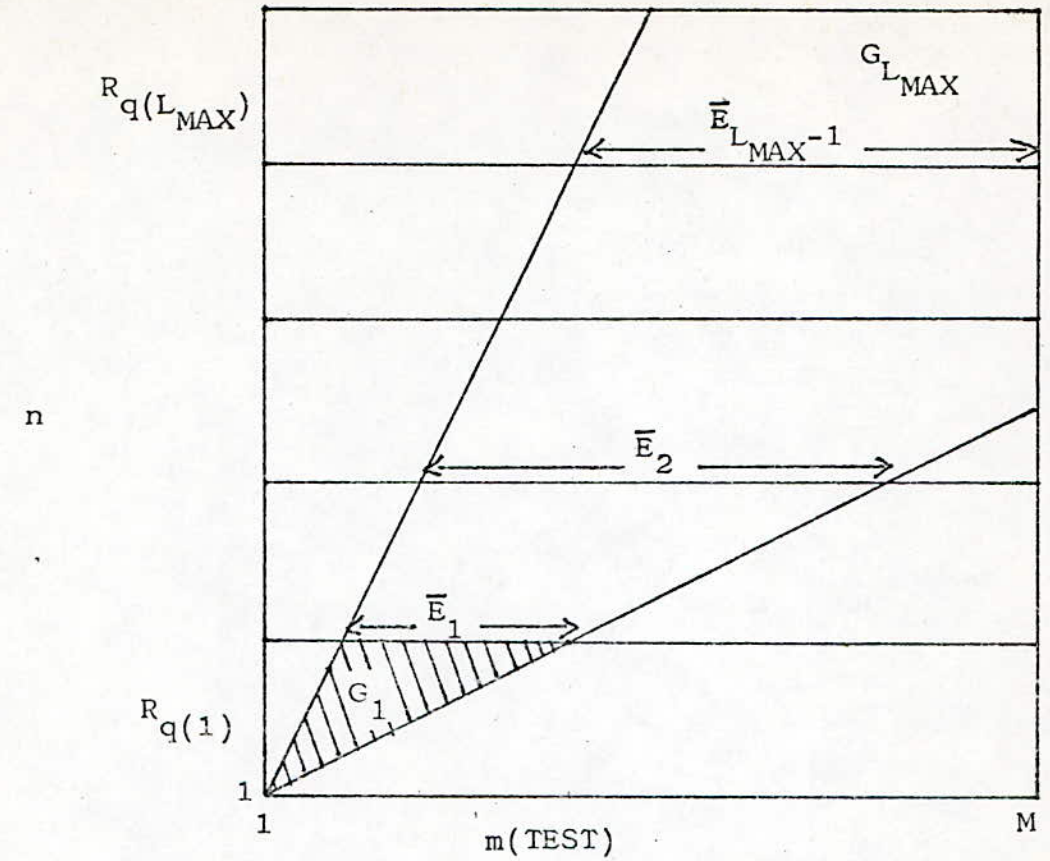


Fig.6. Effet des contraintes aux bornes inférieure et supérieure sur le calcul de l'algorithme LB-DTW

Si on définit $\bar{E}_1'$ comme étant la région finale (le long de l'axe des m) à la fin du premier niveau, alors il sera clair que la largeur de $\bar{E}_1$ variera de la même façon que la longueur des formes de référence.

Généralement, des contraintes locales sur le chemin sont super--posées à la fonction de déformation, pour assurer qu'il ne change pas radicalement quand m augmente. Avec les contraintes d'ITAKURA :

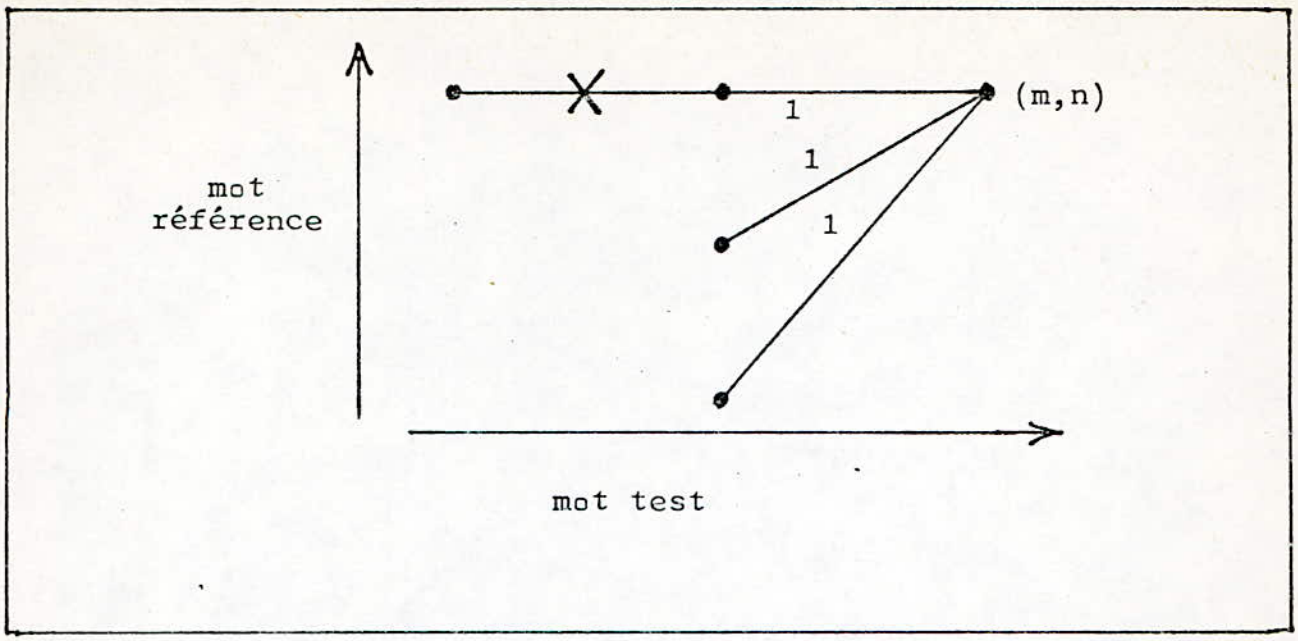
$$\begin{cases} 0 \leqslant w(m) - w(m-1) \leqslant 2 & (18a) \\ 1 \leqslant w(m) - w(m-2) & (18b) \end{cases}$$

les contraintes locales inférieure et supérieure seront :

$$\bar{L}(n) = \max(0, n-2) \qquad (19a)$$

$$\bar{U}(n) = \begin{cases} n & \text{si } w(m-1) \neq w(m-2) \\ n-1 & \text{si } w(m-1) = w(m-2) \end{cases} \qquad (19b)$$

Elles sont illustrées ci-après :



Ces contraintes ont été choisies en raison de l'économie de traite--ment qu'elles réalisent./8/ .

Si on examine la grandeur des régions globales, G_1, au $1^{éme}$ niveau (fig 6), on voit que G_1 augmente continuellement avec 1. Quand 1 se rapproche de L, la grandeur de la chaîne, la région globale G_1 devient plus petite puisque le chemin doit finir au point $(M, \emptyset(L_{MAX}))$, et de nouvelles restrictions L(m) et U(m) peuvent être imposées à la fin du test. Ceci est bien illustré dans la figure 7 qui montre une ligne de pente 1/2, à partir de la fin de la derniére référence, limitant la région supérieure du plan (m,n). Cependant et contrairement à la figure 3, il n'y a pas de ligne de pente 2 à partir du point $(M,\emptyset(L_{MAX}))$, puisqu'il n'est pas connu, à priori, si $L=L_{MAX}$.

Ainsi au lieu de la ligne globale de pente 2 venant de la fin du test, il y a une ligne de même pente à partir de chacun des points finaux possibles $(M,\emptyset(1))$, $1=L_{MIN},\ldots,L_{MAX}$.
Ces restrictions sur le chemin ne sont appliquées qu'au niveau donné, puisque la longueur exacte de la chaîne L n'est pas connue durant le calcul de $D_1(m,n)$. On aboutit alors à une nouvelle forme (convenable) des contraintes aux limites :

$$L(m) = \max\left(\frac{m+1}{2},\ 2(m-M) + \emptyset(1)\right) \qquad (20a)$$

$$U(m) = \min\left(2(m-1)+1,\ \frac{1}{2}(m-M)+\emptyset(L_{MAX})\right). \qquad (20b)$$

où $\emptyset(L_{MAX})$ est la longueur maximale de la super forme de référence concaténée. La région obtenue n'est alors plus un losange (région hachurée).

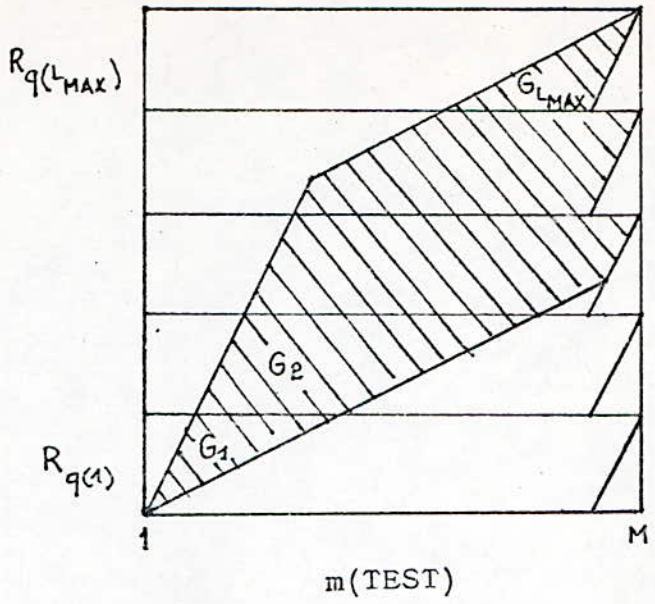


Fig.7. Région globale de l'algorithme LB avec L_{MAX} références possibles.

Il est possible d'éstimer la zone, dans le plan (m,n), recher--chée par l'algorithme LB-DTW.
Pour une chaîne de longueur $L = L_{MAX}$ et une fonction déformation sans contraintes , la zone totale moyenne est :

$$A_T = L_{MAX} \cdot \bar{N} \cdot M \qquad (21)$$

où $\bar{N}$ est la longueur moyenne d'une forme de référence. Alors que dans la région hachurée de la figure 7 la zone totale moyenne réduite est approximativement :

$$A_R = L_{MAX} \cdot \bar{N} \cdot \frac{M}{3} \qquad (22)$$

en admettant que $\bar{N} \approx \frac{M}{L_{MAX}}$ c-à-d que la longueur test est L_{MAX} fois la longueur de référence moyenne.
On remarque donc q'une réduction dans les calculs, d'environ 1/3, est obtenue en utilisant d'une manière appropriée les contraintes aux bornes supérieure et inférieure.

IV.6.2 Techniques de réduction d'intervalle

Plusieurs réductions de calcul peuvent être apportées à l'algorithme de LB. Une des plus simples est de réduire l'intervalle sur m à chaque niveau, comme illustré sur la figure 8. La partie (a) de cette figure montre la région originale (bornée par les lignes dis--continues L et U) et la région réduite (bornée par les traits pleins)

L'intervalle est réduit telles que les régions du plan (m,n) où les chemins ont déjà accumulé une grande distance, ne sont pas considérées comme des points de début potentiels pour le niveau suivant. Quant à la partie (b) , elle illustre la méthode utilisée pour réduire l'inter-valle cherché.

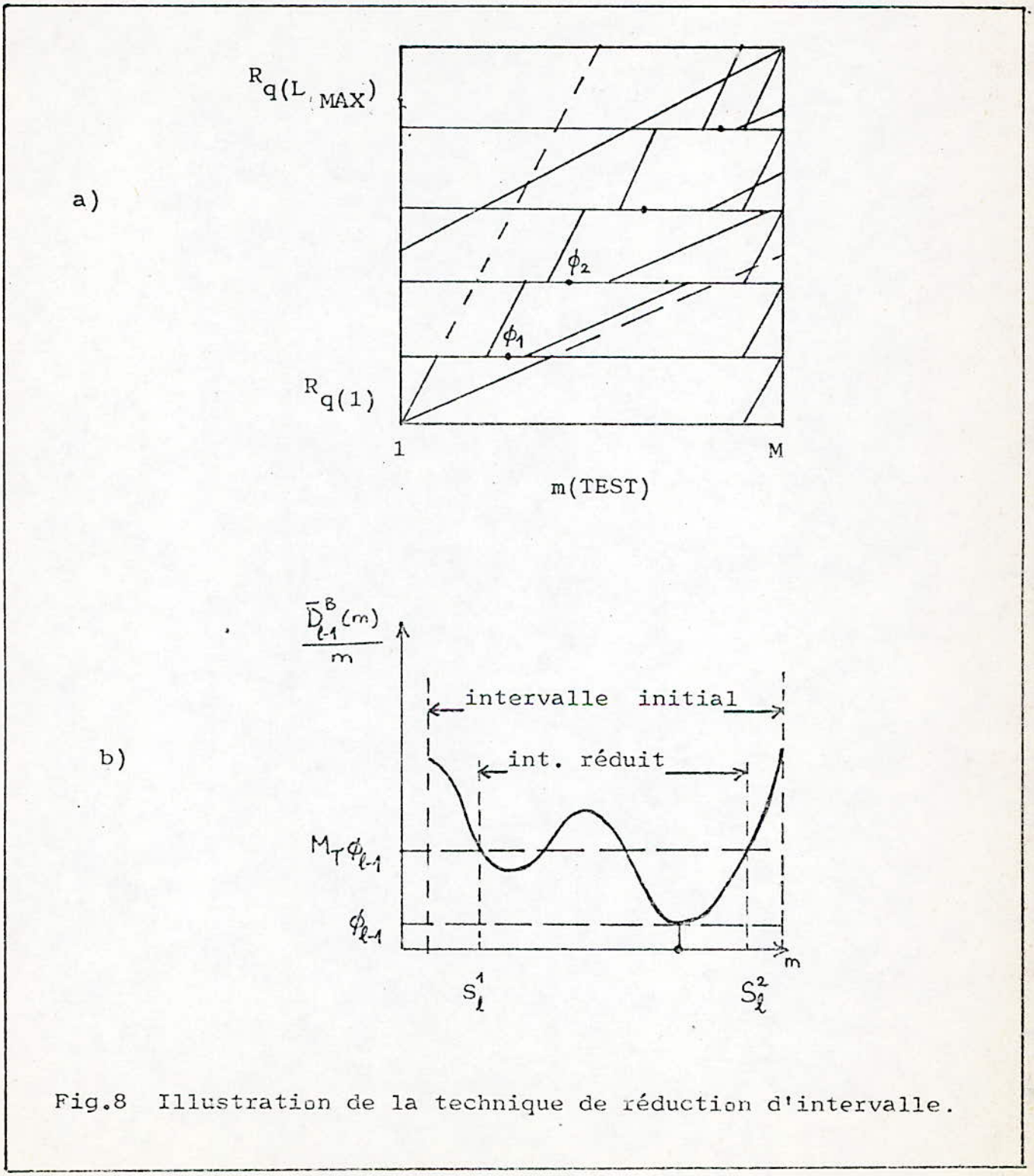


Fig.8 Illustration de la technique de réduction d'intervalle.

Au début du $l^{éme}$ niveau, la distance moyenne minimale à la fin du (l-1)ème niveau est donnée par $\emptyset_{l-1}$:

$$\emptyset_{l-1} = \min_{1 \leq m \leq M} \left(\frac{\bar{D}^B_{l-1}(m)}{m} \right) \qquad (23)$$

Deux valeurs S_l^1 et S_l^2 peuvent alors être trouvées comme suit :

S_l^1 = le plus grand m tel que :

$$\left(\frac{\bar{D}^B_{l-1}(m)}{m} \right) > \emptyset_{l-1} \cdot M_T \quad \text{pour tout } m < S_l^1 .$$

S_l^2 = le plus petit m tel que :

$$\left(\frac{\bar{D}^B_{l-1}(m)}{m} \right) > \emptyset_{l-1} \cdot M_T \quad \text{pour tout } m > S_l^2 .$$

où M_T est un paramètre contrôlant la grandeur de l'intervalle de début. Quand M_T approche la valeur 1, les valeurs S_l^1 et S_l^2 se rapprochent l'une de l'autre et la largeur de l'intervalle réduit tend vers un point unique. Ceci est équivalent à faire une décision sans équivoque au sujet du choix de la référence correcte à la fin du (l-1) ème niveau. Quand M_T tend vers l'infini, l'intervalle réduit est le même que l'intervalle original.
La figure 8 (b) illustre l'importance d'utiliser une valeur raisonnable de M_T (généralement $M_T \approx 1,2$). Dans cet exemple, la courbe de $\left(\frac{\bar{D}^B_{l-1}(m)}{m} \right)$ a deux minima distincts, ceci représente généralement deux régions de fin possibles pour la (l-1) ème référence, et n'importe quelle réduction d'intervalle qui élimine l'une d'elles pourrait potentiellement conduire à des erreurs dans l'obtention de la meilleure chaîne. Cet exemple illustre aussi l'importance de chercher S_l^1 et S_l^2 à partir des extrémités de l'intervalle puisque la courbe excède le seuil $(M_T . \emptyset_{l-1})$ entre S_l^1 et S_l^2. Dans ce cas, on ne veut pas réduire l'intervalle pour éliminer cette région.

Il est donc utile d'insérer la technique de réduction d'intervalle dans l'algorithme 5, en remplaçant l'étape 4) par ce qui suit :

4) Pour m = 0,1,...,M initialiser

$$D_l'(m,0) = \begin{cases} \bar{D}^B_{l-1}(m) & \text{si } S_l^1 \leq m \leq S_l^2 \\ \infty & \text{ailleurs} \end{cases}$$

$$F_1(m,0) = m \qquad \text{si } S_1^1 \leqslant m \leqslant S_1^2$$

où $S_1^1 = S_1^2 = 1$

En outre les fonctions de bornes modifiées supérieure et inférieure $U_1(m)$ et $L_1(m)$ sont données par :

$$L_1(m) = \max\left(1, \frac{(m-S_1^2)}{2}, 2(m-M) + N_v\right) \qquad (24a)$$

$$U_1(m) = \min\left(N_v, 2(m-S_1^1), \frac{m-M}{2} + (L_{MAX}-1)N_{MAX}+N_v\right) \qquad (24b)$$

où N_{MAX} est la longueur de la forme de réf la plus longue.

Le dernier terme dans (24b) reflète une ligne de pente 1/2 qui vient du point m=M (comme dans la figure 7).

Une autre méthode de réduction d'intervalle est illustrée dans la figure 9. L'intervalle est réduit non seulement par rétrécissement de la région de début (axe des m) mais aussi en suivant le minimum local. Dans ce cas, la distance locale accumulée $D_1(m,n)$ est calculée pour :

$$c(m)-\varepsilon \leqslant n \leqslant c(m)+\varepsilon \qquad 1 \leqslant m \leqslant M$$

où

$$c(m) = \underset{c(m-1)-\varepsilon \leqslant n \leqslant c(m-1)+\varepsilon}{\operatorname{argmin}} \left(D_1(m-1,n)\right)$$

$$c(1) = 1$$

La borne inférieure doit toujours inclure un point de début valable, s'il y'en a un, c-à-d $c(m)-\varepsilon$ doit être mis à 1 si $S_1^1 \leqslant m \leqslant S_1^2$. Cette contrainte est nécessaire pour prévenir la perte éventuelle de points de début de chemins possibles. Il faut noter que ε peut être une variable adaptée à la donnée, c-à-d ε est grand quand la distance le long d'une bande verticale est proche du minimum local sur un large intervalle, et ε est petit quand la même distance s'éloigne rapide--ment du minimum local.

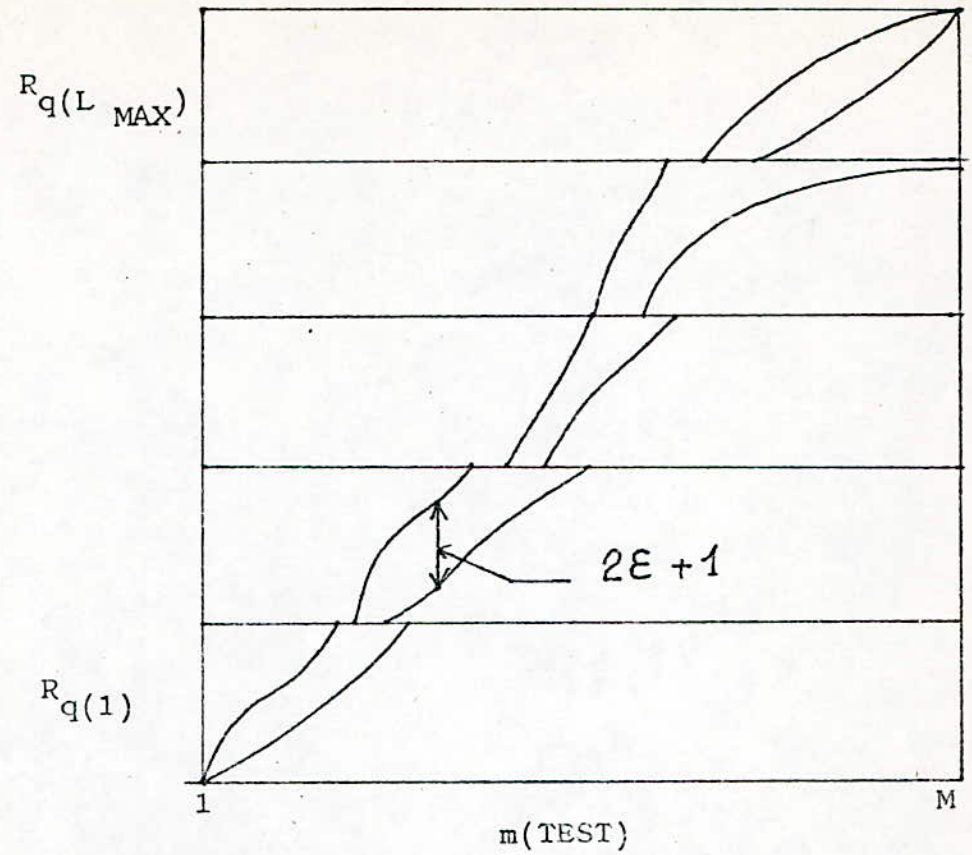


Fig.9 Illustration de la réduction d'intervalle(de n) .

Il est aussi possible d'avoir une nouvelle réduction dans le calcul, par l'utilisation de seuils de distances accumulées. Un seuil de distance est utilisé pour abandonner la comparaison, pour une forme de référence donnée R_v, à un niveau donné l, quand la distance accumulée à ce niveau excède ce seuil. On définit la moyenne de distance en (m,n) au l éme niveau comme suit :

$$\Delta D_l(m,n) = \frac{D_l(m,n) - \overline{D}^{B}_{l-1}\,(F_l(m,n))}{m - F_l(m,n)}$$

Et à chaque fois que $\Delta D_l(m,n)$ excède un seuil de la forme :

$$\hat{T}(m) = T_{MIN} + T_{MAX}\,(m - F_l(m,n)),$$

la comparaison pour la référence courante est abandonnée. Cependant ce seuil ne doit être appliqué qu'une fois tous les points de début essayés. c-à-d pour $m > S_1^2$.

IV.6.3 Elimination de la contrainte sur le dernier point du test

Plutôt que d'obliger le chemin à finir à la trame M du test ($\emptyset$ (L) = w(M)), une plus grande flexibilité est obtenue si on lui permet de finir à l'intérieur de la région comprise entre $M-\delta_{END}$ et M(si $\delta_{END}=0$ il est clair qu'on est dans le cas de la contrainte sur le point final). L'utilisation d'une région de fin variable est importante car elle facilite la tâche à la machine(le reconnaisseur) dans la détection de la trame finale de la chaîne parlée de mots.

Cette modification peut être introduite dans l'algorithme de LB , en cherchant la meilleure chaîne (correspondant à la plus petite distance normalisée) à l'intérieur de la région de fin modifiée. Ceci en remplaçant dans l'algorithme 5 (IV.4), les étapes 8) et 9) par :

$$8') \quad (L,E) = \underset{L_{MIN} \leq l \leq L_{MAX}}{\text{argmin}} \quad \underset{M-\delta_{END} \leq e \leq M}{\text{argmin}} \quad (\bar{D}_1^B(e) / e)$$

$$9) \quad e_L = E$$

où E est utilisé pour enregistrer le point de la fin.

IV.6.4 Formes de référence modifiées (introduction des flous)

Une autre modification de l'algorithme de LB est suggérée par RABINER et SCHMIDT. Ils proposent que la forme de référence ne soit pas utilisée entièrement, mais qu'il soit permis d'éliminer une partie de sa fin. Ce type de modification rend compte du phénomène de coarticulation entre des formes de référence consécutives. Une autre interprétation est que les formes de référence, obtenues à partir de mots isolés , sont généralement plus longues que le mot parlé dans une chaîne.

Cette modification est introduite en exigeant qu'à chaque niveau, la fonction de déformation finisse quelque part entre les trames $N_v-\delta_{R_2}$ et N_v de la forme de référence, plutôt que d'être forcée de finir à la trame N_v (il est clair que si $\delta_{R_2}=0$, cette modification devient transparante dans l'algorithme). En général, le nombre maximum de trames δ_{R_2} est une fonction de la forme de référence. Cependant et afin de simplifier, on admet que δ_{R_2} est constante pour toutes les formes de référence. L'étape 6) de l'algorithme 5 est ainsi modifiée comme suit :

6) Pour m = 1,2,...,M calculer

$$\bar{n} = \underset{N_v - \delta_{R_2} \leq n \leq N_v}{\operatorname{argmin}} \; (D_1(m,n))$$

$$\bar{D}_1^v(m) = D_1(m,\bar{n})$$

$$\bar{F}_1^v(m) = F_1(m,\bar{n})$$

où $\bar{n}$ est utilisé pour indiquer le meilleur point final dans la forme de référence.

Une modification similaire peut être utilisée au début des formes de référence. Si on admet que la fonction de déformation peut commencer entre la trame 1 et la trame $1 + \delta_{R_1}$ de la forme de référence (quand $\delta_{R_1} = 0$, toute la région de début est utilisée), alors il est nécessaire de modifier l'étape 5 de l'algorithme 5.

5) Pour $m = 1,2,\ldots,M$ et pour $n = L_1(m),\ldots,U_1(m)$, calculer en utilisant $R = R_v$

$$\hat{n} = \begin{cases} \underset{n'=0, \text{ou } \bar{L}(n) \leq n' \leq \bar{U}(n)}{\operatorname{argmin}} \; (D_1(m-1,n')) & \text{pour } 1 \leq n \leq 1+\delta_{R_1} \\ \underset{\bar{L}(n) \leq n' \leq \bar{U}(n)}{\operatorname{argmin}} \; (D_1(m-1,n')) & \text{pour } n > 1+\delta_{R_1} \end{cases}$$

$$D_1(m,n) = d_1(m,n) + D_1(m-1,\hat{n})$$

$$F_1(m,n) = F_1(m-1,\hat{n})$$

Les contraintes locales modifiées disent que si n est dans la région de début " dilatée " de la forme de référence (dans les $(1+\delta_{R_1})$ premières trames), alors la distance accummulée antérieure peut provenir des conditions initiales à la fin du niveau antérieur.

Les effets des variables δ_{R_1}, δ_{R_2}, δ_{END}, M_T, , , sur les chemins de Level Building sont illustrés sur la figure 10 .

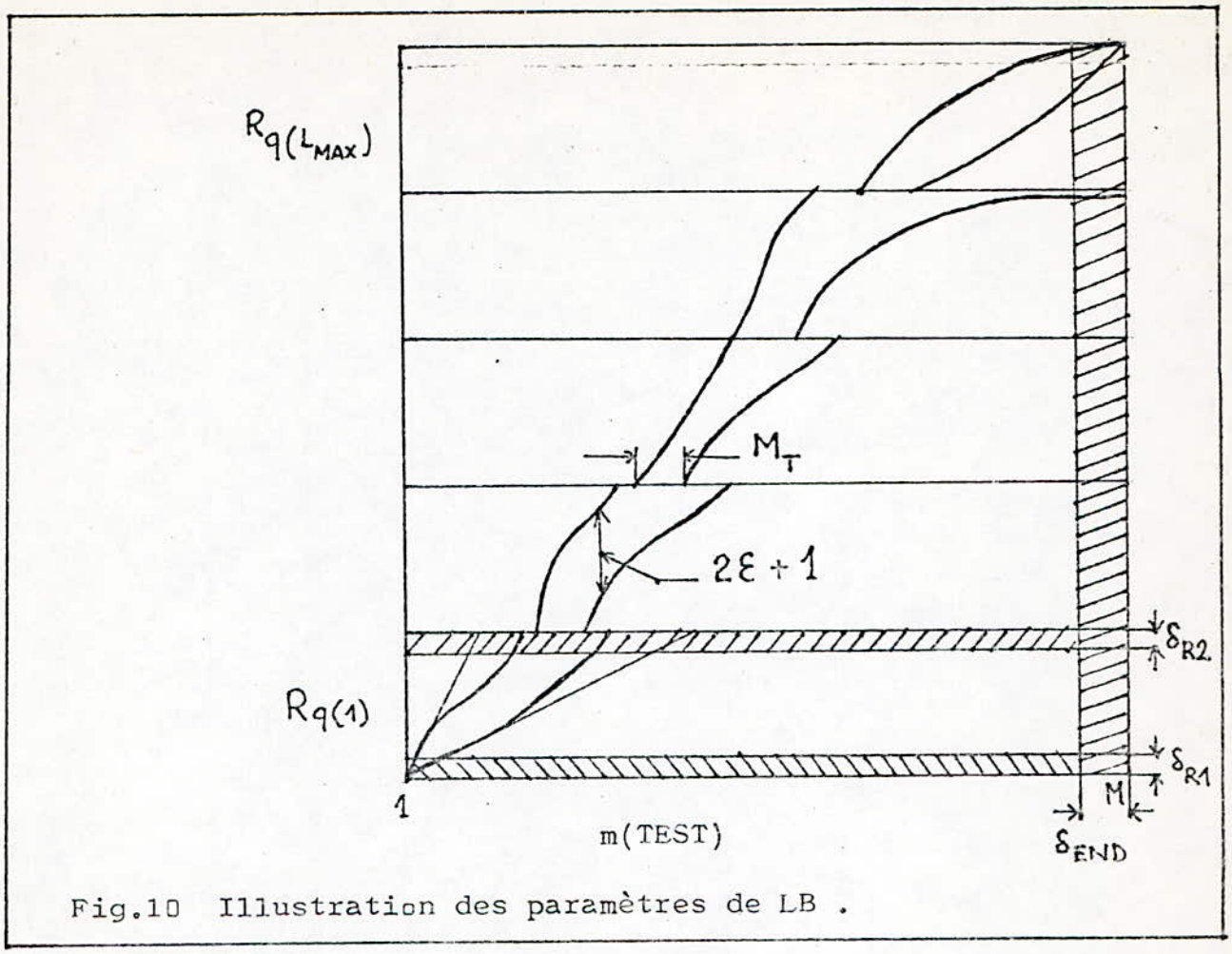


Fig.10 Illustration des paramètres de LB .

IV.6.5. Chaînes candidates multiples

Pour plusieurs applications de reconnaissance de mot connecté, il est exigé que plus d'une super forme de référence soit donnée comme candidate de reconnaissance possible. Comme il a été décrit, l'algorithme de LB ne permet pas d'avoir directement un groupe de super formes de référence, car seule l'information concernant la meilleure candidate est retenue. Une manière raisonnable de générer une liste de candidates plausibles est de retenir les données (dis--tances, pointeurs arrière, etc..) sur le meilleur et le second choix à chaque niveau et en chaque point final.

Pour en obtenir d'autres, la procédure normale de backtracking est suivie avec la différence que les deuxièmes meilleures références sont insérées à la place des premières aux différents niveaux. Cependant, il faut noter que cette technique donne une liste raison--nable de candidates sans toutefois qu'elles soient les meilleures.

IV.7. L'algorithme LB-DTW modifié

Après avoir présenté les modifications susceptibles d'amélio-rer l'algorithme LB-DTW (algorithme 5) , nous en proposons un qui satisfait certaines d'entre elles.

1) Initialiser $\bar{D}_0^B(m) = 0$ pour $m = 0$

$\bar{D}_0^B(m) = \infty$ pour $m = 1,2,\ldots,M$

$\bar{D}_1^B(0) = \infty$ pour $1 = 1,2,\ldots,L_{MAX}$

2) Pour $1 = 1,2,\ldots,L_{MAX}$ faire les étapes de 3) à 7)

3) Pour $v = 1,2,\ldots,V$ faire les étapes de 4) à 6)

4) Pour $m = 0,1,\ldots,M$ initialiser

$$D_1(m,0) = \begin{cases} \bar{D}_{1-1}^B(m) & \text{si } S_1^1 \leq m \leq S_1^2 \\ \infty & \text{ailleurs} \end{cases}$$

$$F_1(m,0) = m \quad \text{si } S_1^1 \leq m \leq S_1^2$$

où $S_1^1 = S_1^2 = 1$

5) Pour $m = S_1^1, S_1^1+1,\ldots,M$ et pour $n = L_1(m),\ldots,U_1(m)$, calculer en utilisant $R = R_v$

$$\hat{n} = \begin{cases} \underset{n'=0, \text{ ou } \bar{L}(n)\leq n'\leq \bar{U}(n)}{\text{argmin}} & (D_1(m-1,n')) \quad \text{pour } 1 \leq n \leq 1 + \delta_{R_1} \\ \underset{\bar{L}(n)\leq n'\leq \bar{U}(n)}{\text{argmin}} & (D_1(m-1,n')) \quad \text{pour } n > 1 + \delta_{R_1} \end{cases}$$

$$D_1(m,n) = d_1(m,n) + D_1(m-1,\hat{n})$$

$$F_1(m,n) = F_1(m-1,\hat{n})$$

6) Pour $m = S_1^1, S_1^1+1,..,M$ calculer

$$\bar{n} = \underset{N_v - \delta_{R_2} \leq n \leq N_v}{\operatorname{argmin}} \; (D_1^v(m,n))$$

$$\bar{D}_1^v(m) = D_1^v(m,\bar{n})$$

$$\bar{F}_1^v(m) = F_1^v(m,\bar{n})$$

7) Pour $m = S_1^1, S_1^1+1, \ldots, M$, calculer

$$\bar{D}_1^B(m) = \min_{1 \leq v \leq V} (\bar{D}_1^v(m))$$

$$W_1(m) = \underset{1 \leq v \leq V}{\operatorname{argmin}} \; (\bar{D}_1^v(m))$$

$$\bar{F}_1^B(m) = \bar{F}_1^{W_1(m)}(m)$$

8) $$(L,E) = \underset{L_{MIN} \leq l \leq L_{MAX}}{\operatorname{argmin}} \; \underset{M-\delta_{END} \leq e \leq M}{\operatorname{argmin}} \; (\bar{D}_1^B(e)/e)$$

9) $e_L = E$ (E est utilisé pour enregistrer le point de la fin)

10) $e_l = \bar{F}_{l+1}^B(e_{l+1})$, $l = L-1, L-2, \ldots, 1.$

11) $\bar{R}^s = R_{W_1}(e_1) \oplus R_{W_2}(e_2) \oplus \ldots \oplus R_{W_L}(e_L)$.

Remarque sur l'algorithme proposé :

Cet algorithme ne peût être dit optimal puisque n'utilisant pas toutes les règles susceptibles de l'optimiser, telles que les règles syntaxiques qui nécessiteraient un apprentissage plus ou moins long.

IV.8. Comparaisons d'algorithmes DTW du mot connecté.

Ces comparaisons concernent les algorithmes LB-DTW, LB-DTW modifié et le Two-Level de SAKOE. Elles sont basées sur le calcul (ou nombre d'opérations) et le stockage dans la reconnaissance d'une chaîne de L digits .

IV.8.1. Calcul

Algorithme LB

Les contraintes locales et globales obligeant le chemin de déformation à se trouver à l'intérieur du parallélogramme de la figure7, le nombre d'opérations de calcul de distance locale est :

$$ND_{A5} = V.L_{MAX} . \overline{N} . M/3$$

où $\overline{N}$ est la durée moyenne (en trames) d'une forme de référence

L_{MAX} est le nombre de niveaux (si $L < L_{MAX}$, donc génralement L_{MAX} niveaux ne seront pas exigés)

V est le nombre de références par niveau

M est la durée du test

$\overline{N}$ .M/3 est le nombre moyen des distances à chaque niveau

Algorithme Two-Level de SAKOE

Le nombre d'opérations est :

$$ND_{S1} = V.\overline{N} . M.(2R+1)$$

où R est un paramètre d'intervalle/12/.

Algorithme LB-DTW modifié

En incorporant les techniques de réduction d'intervalle dans l'algorithme 5, le nombre d'opérations de calcul de distance locale devient :

$$ND_{A5R}=V.L.\overline{N}\ (2\mathcal{E}+1)$$

où $\mathcal{E}$ est le paramètre d'intervalle (minimum local)

IV.8.2 Stockage

Le stockage pour les différents algorithmes DTW du mot connecté concernent les réseaux $(\overline{D}_1^{\,B}(m),\ W_1(m),\ \overline{F}_1^{B}(m))$ spécifiques à l'algo-rithme.

Algorithme LB (modifié ou non)

Chacun des 3 vecteurs est de taille M, de là le stockage est :

$$S_{A5}=S_{A5R}=3.M.L_{MAX}$$

Algorithme Two-Level de SAKOE

$$S_{S1}=2.M.(2R+1)$$

IV.8.3 Tableaux de comparaison

On regroupe les résultats dans un tableau ;

	Level Building	Level Building modifié	Two - Level de (SAKOE)
Nbre d'opérations de calcul de distances	$V.L_{MAX}.\overline{N}.M/3$	$V.L.\overline{N}(2\mathcal{E}+1)$	$V.M.\overline{N}.(2R+1)$
Stockage	$3.M.L_{MAX}$	$3.M.L_{MAX}$	$2.M.(2R+1)$

Le deuxième tableau,ci-dessous,donne une comparaison numérique pour les valeurs de paramètres suivantes :

$L_{MAX} = 5$, $V = 10$, $M = 120$, $\overline{N} = 35$, $\mathfrak{C} = 12$, $R = 12$, $L = 4$.

	LB	LB MODIFIE	Two - Level (SAKOE)
Nbre d'opérations de calcul de distances	70.000	35.000	1 050 000
Stockage	1.800	1.800	6.000

Ces valeurs numériques justifient,encore une fois,le choix de l'algorithme LB :Le Two-Level de SAKOE exige 3 fois plus de capacité de stockage et 15 fois plus d'opérations.D'autre part,on voit que le calcul a été réduit de moitié en modifiant l'algorithme LB .

V. CONCLUSION

Nous venons donc de voir, l'algorithme permettant de réaliser l'alignement DTW d'une chaîne de mot connecté et d'un groupe de formes de référence concaténées, utilisant pour cela, une fonction asymétrique et des formes de référence concaténées indépendantes (pas d'interaction de niveaux).La sortie finale de l'algorithme étant la super forme de référence, adaptant au mieux la chaîne d'entrée. Un nombre de modifi--cationsa été alors apporté à l'algorithme afin d'augmenter la flexibi--lité et l'exactitude de cette méthode.

Le domaine de la parole est tellement vaste que nous fûmes obli--gés, malgré nous, " d'éviter " des domaines qui lui sont pourtant bien liés, telles que la phonétique, l'analyse (codage)... auquel cas il ne resterait plus qu'à implanter l'algorithme sur un processeur de traitement rapide (temps réel).

Ainsi nous avons découvert une voie qui a uni beaucoup de chercheurs. Le but est d'exploiter " la troisiéme main " de l'homme (que représente sa voix) en lui laissant libres les deux autres. De profonds boulversements sont donc attendus avec l'introduction des techniques vocales dans l'informatique, l'industrie, la télématique, la bureautique,... Les applications actuelles et potentielles, multiples, le confirment :

- Un robot à entrée, sortie vocales,
- Un composeur téléphonique vocal
- La programmation vocale des machines
- La machine à écrire à dictée vocale
- La saisie de données

En ce qui concerne l'handicapé, auquel les chercheurs n'ont jamais cessé de penser, il peut contrôler son environnement par sa voix (allumer un poste de TV, déclencher une alarme), et commander son fauteuil roulant, son lit...

* * * * * *

BIBLIOGRAPHIE

/1/.B.BOUSSEKSOU, Reconnaissance automatique de la parole par la méthode globale.Application aux particularités linguistiques de l'arabe standard.Thèse de Magister,1983.

/2/.E.CATIER, La parole:Analyse-Synthèse-Reconnaissance.TLE Décembre 1983,n°489,pp 16-25.

/3/.A.FELDBAUM, Principes théoriques des systèmes asservis opti--maux.Editions MIR. MOSCOU,1973.

/4/.C.GAGNOULET,G.MERCIER, Le point sur la reconnaissance de la parole.L'écho des Recherches,Juillet 1981.

/5/.J.GUIBERT, La parole:compréhension et synthèse par les ordi--nateurs.Presses Universitaires de France,1979.

/6/.Haut-Parleur(revue) n°1723 Décembre 1985.

/7/.M.KUNT, Traitement numérique des signaux.Editions Georgi , Dunod,1981.

/8/.A.MENACER, Reconnaissance de la parole en mode multilocuteur par des méthodes globales(mots isolés),Février 1985.

/9/.C.S.MYERS,L.R.RABINER,A level building dynamic time warping algorithm for connected word recognition.IEEE Trans ASSP , April 1981,Vol29,n°2,pp 284 -297.

/10/.C.S.MYERS,L.R.RABINER, Connected digit recognition using a level buildihg DTW algorithm.IEEE Trans ASSP,June 1981,Vol 29,n°3,pp 351-363.

/11/.L.R.RABINER,S.E.LEVINSON, Isolated and connected word reco--gnition.Theory and selected applications.IEEE Trans Comm , May 1981,Vol 29,n°5,pp 639-644.

/12/.H.SAKOE,S.CHIBA, Dynamic programming algorithm optimization for spoken word recognition.IEEE Trans ASSP,February 1978 , Vol 26,n°1,pp 43-49.

/13/.R.W.SCHAFER,L.R.RABINER, Digital processing of speech signals. Prentice Hall, 1978.

/14/.Sciences et Techniques(revue). L'ordinateur prend la parole . Reconnaissance et synthèse vocales ,n°2,Hors série,1984.

/15/.Z.SIAD,R.HAMDI, Etude de programme pour la reconstitution de la parole par prédiction linéaire.Projet de fin d'études . Janv 84.

/16/.J.T.TOU,R.C.GONZALEZ, Pattern recognition principles.Addison-Wesley Publishing Company.Massachusetts,1974.