الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

pour l'obtention du diplôme d'Ingénieur d'Etat

SUJET

RECONNAISSANCE DE LA PAROLE PAR LA METHODE DES CHAINES DE MARKOV CACHEES

Proposée par : **Mr BOUSSEKSOU** - Etudié par : **T. BERBAR** - Dirigée par : **Mr BOUSSEKSOU**

PROMOTION : JUIN 1991

E.N.P., 10, avenue Hacène Badi - EL HARRACH - ALGER

الجمعورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

pour l'obtention du diplôme d'Ingénieur d'Etat

SUJET

RECONNAISSANCE DE LA PAROLE PAR LA METHODE DES CHAINES DE MARKOV CACHEES

Proposée par : **Mr BOUSSEKSOU** - Etudié par : **T. BERBAR** - Dirigée par : **Mr BOUSSEKSOU**

PROMOTION : JUIN 1991

E.N.P. , 10, avenue Hacène Badi - EL HARRACH - ALGER

Dedicaces

Je dedie ce modeste travail a ma mère, mon père et toute ma famille .

Je pense aussi à ceux et celles qui sont mes amis .

Tarik

REMERCIEMENTS

Je tient à remercier tous ceux qui m'ont aidé et encouragé durant mes dernières années d'étude.

Je remercie vivement M^r^ Bousseksou, mon promoteur, qui a su me guider et me conseiller pour mener ce projet à son terme.

J'estime que c'est grâce à mon oncle Djamel que je peux présenter ce polycopié à temps, aussi je lui suis très reconnaissant.

Je remercie ,de même, le personnel de la bibliothèque et du centre de calcul, pour leur amabilité et toute l'aide précieuse et irremplaçable qu'ils m'ont fournie.

Et bien sûr, je n'oublie pas mes amis. Car qu'ils soient de Polytechnique, de l'Epau, ou de tout autre horizon, ils ont toujours prétés une oreille attentive à mes questions et même à mes suggestions.

TABLE DES MATIERES

BIBLIOGRAPHIE

[1] M.BOITE :Traitement de la parole

[2] Traitement du signal (Revue n°5, 1988)
"Champs aléatoires de Pickard"

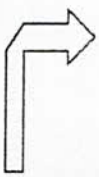
[3] F.CINARE : Reconnaissance et synthèse de la parole

[4] E.EMERIT :Cours de phonétique acoustique

[5] J.S.LIENARD :Les processus de la communication parlée

[6] J.LIFERMANN : Les méthodes rapides de transformation du signal

Chapitre I



INTRODUCTION

La reconnaissance automatique de la parole est un domaine en plein essor, grâce à l'apparition de processeurs spécialisés très performants et d'algorithmes très efficaces.
Déjà, on peut trouver sur le marché des produits aux capacités limitées certes, mais pouvant rendre de grands services aux utilîsateurs potentiels de cette technologie.
Rappelons que les objectifs poursuivis sont un codage optimal du signal vocal pour sa transmission et son stockage, et le dialogue homme-machine sans l'intermediaire de console.
Tout naturellement on préfère extraire les paramètres les plus pertinents du signal vocal, aussi les diverses recherches ont montré l'intérêt du codage par prédiction linéaire (LPC).
En reconnaissance proprement dite, on dispose de plusieurs algorithmes exploitant les paramètres de l'analyse.
L'algorithme DTW de programmation dynamique permet une normalisation temporelle de la durée d'un mot pour optimiser la reconnaissance.
En revanche la méthode stochastique spécule sur la probabilité de production d'un mot par un automate stochastique, et c'est l'objet principal de notre étude.

CHAPITRE 1 :

PRODUCTION ET MODELISATION DU SIGNAL VOCAL

La parole est la faculté de communiquer la pensée par un système de sons articulés.
L'information d'un message parlé réside dans les fluctuations de la pression de l'air engendrées, puis émises, par l'appareil phonatoire.

1.1-MECANISME DE LA PHONATION

La parole est le résultat de l'action volontaire et coordonnée des appareils respiratoires et masticatoires.
L'appareil respiratoire fournit l'énergie nécessaire lorsque l'air est expiré par la trachée artère. Au sommet de celle-ci se trouve le larynx où la pression de l'air est modulée avant d'être appliquée au conduit vocal qui s'étend du pharynx jusqu'aux lèvres. C'est un ensemble de cavités qu'on peut distinguer sur la figure 1: la cavité pharyngienne,la cavité buccale et en dérivation la cavité nasale.

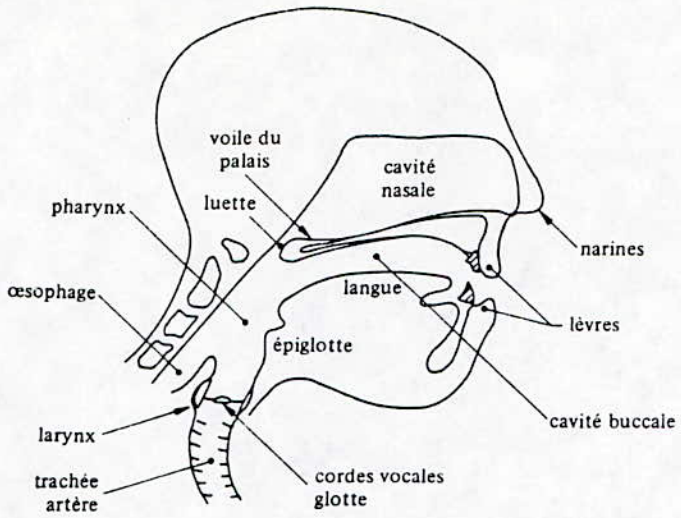


Fig-1 Appareil phonatoire

Le conduit vocal peut être considéré comme une succession de tubes ou cavités acoustiques de reactions diverses.

Les *sons voisés* résultent donc de l'excitation du conduit vocal par des impulsions périodiques de pression liées aux oscillations des cordes vocales: l'ouverture brusque de la glotte libère la pression accumulée en amont; elle se referme ensuite plus graduellement.

Un son voisé est un signal quasi périodique (fig 2et3)

Sur la figure 3 on observe les raies qui correspondent aux harmoniques du fondamental F_o. L'enveloppe de ces raies présente des maximums appelés *formants* et qui correspondent aux fréquences propres F_i ($i=1,2,3...$) du conduit vocal.

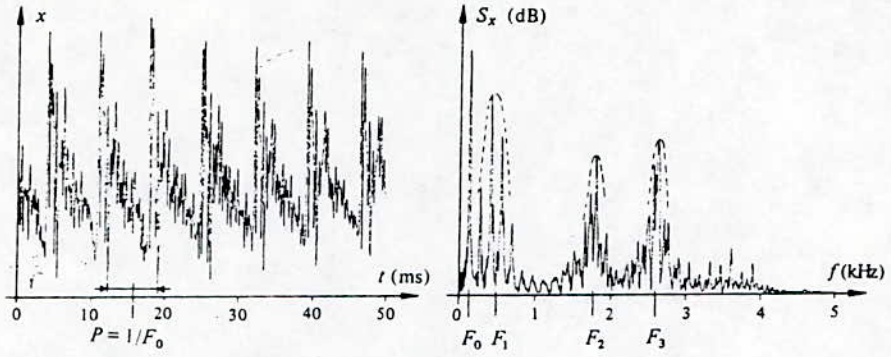


Fig 2 et 3:Un signal vocal et son spectre

Les trois premiers formants sont essentiels pour caractériser le spectre vocal, les formants d'ordres supérieurs ont une influence limitée.

Un son non voisé ne présente pas de structure périodique, il peut être considéré comme un bruit blanc filtré par la transmittance du conduit vocal. Remarquons toutefois qu'un son non voisé présente la même structure formantique que le même son voisé.

1.2-MODELISATION DE LA PRODUCTION DE LA PAROLE

La modélisation proposée ici utilise le formalisme des systèmes echantillonnés (transformée en Z)

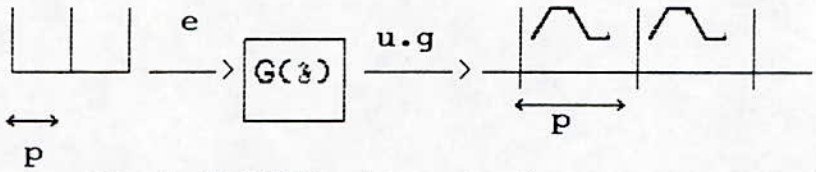


Fig-4 Modélisation de la source pour les sons voisés

Pour les sons voisés, la source est un train périodique d'onde (fig 4). Ce train d'onde est modélisé par un filtre passe bas d'ordre 2 à pôles réels dont la fréquence de coupure est de l'ordre de 100 Hz :

$$G(z) = \frac{A}{(1+\alpha . z^{-1})(1+\beta . z^{-1})}$$

Pour les sons non voisés, la source est un bruit blanc.
On peut assimiler le conduit vocal à une cascade de resonateur dont la transmittance est de la forme:

$$V(z) = \frac{B}{\prod_{k=1}^{M} (1+b_{1k} . z^{-1} + b_{2k} . z^{-2})}$$

Chaque résonateur correspond à un formant dont la fréquence centrale est donnée par:

$$F_k = (1/(2.\pi))\ f_e.\cos^{-1}\left[\frac{-b_{1k}}{2.\sqrt{b_{2k}}}\right]$$

f_e : fréquence d'échantillonnage.

Le son est finalement émis à travers l'ouverture des lèvres, où:

$$R(z) = C\ (1-z^{-1})$$

En résumé, la transmittance globale entre le train d'impulsion et le son émis:

$$T(z)=G(z).V(z).R(z)$$

$$= \frac{\sigma.(1-z^{-1})}{(1+\alpha.z^{-1}).(1+\beta.z^{-1}).\prod_{k=1}^{M}(1+b_{1k}.z^{-1}+b_{2k}.z^{-2})}$$

On suppose que l'un des pôles de $G(z)$ est proche de l'unité donc:

$$T(z) = \frac{\sigma}{(1+\alpha.z^{-1}).\prod_{k=1}^{M}(1+b_{1k}.z^{-1}+b_{2k}.z^{-2})} = \frac{\sigma}{A(z)}$$

On a posé: $A(\mathfrak{z}) = (1+\alpha.\mathfrak{z}^{-1}).\prod_{k=1}^{M}(1+b_{1k}.\mathfrak{z}^{-1}+b_{2k}.\mathfrak{z}^{-2})$

$$= 1+\sum_{i=1}^{2M+1} a_i.\mathfrak{z}^{-i}$$

La transmittance de ce modèle est dite *tous-pôles*; son inverse le polynôme $A(\mathfrak{z})$ est la transmittance du filtre inverse. Ses limitations sont cependant évidente. En premier lieu, la source est soit un train périodique d'impulsions, soit un bruit blanc; les sons fricatifs voisés ($v,\mathfrak{z},...$) ne peuvent pas être produit par ce modèle.
En second lieu,la production de *sons nasalisés* fait intervenir deux cavités associées en parallèle; la transmittance correspondante est de la forme:

$$\frac{\sigma_1}{A_1(\mathfrak{z})} + \frac{\sigma_2}{A_2(\mathfrak{z})} = \frac{\sigma_1.A_2(\mathfrak{z}) + \sigma_2.A_1(\mathfrak{z})}{A_1(\mathfrak{z}).A_2(\mathfrak{z})}$$

Toutefois, malgrés ses limitations la transmittance tous-pôles est la base de la modélisation par prédiction linéaire.
Pour pouvoir assimiler le numérateur à une constante,on doit surestimer le degré du dénominateur.
On remarque aussi que le signal vocal est stationnaire pendant des intervalles de temps de l'ordre de 20ms.
Durant cet intervalle les coefficients de $T(\mathfrak{z})$ sont constants.

1.3-LES PHONEMES

La plupart des langues naturelles sont composées à partir de sons distincts,les phonèmes.

Un phonème est la plus petite unité présente dans la parole et susceptible par sa présence de changer la signification d'un mot.
La production d'un phonème donné laisse toutefois place à une certaine variabilité sur le plan acoustique.
Sur la figure 4, on peut voir une classification des phonèmes.

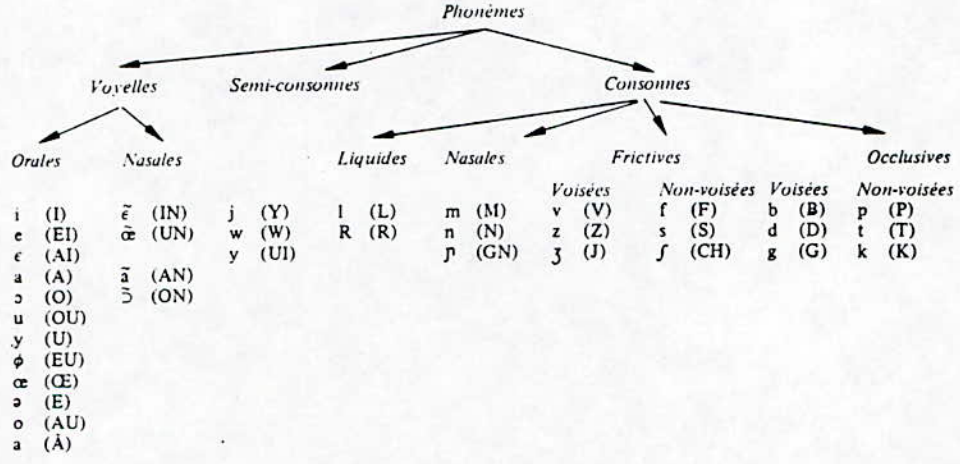


Fig-4 Phonèmes de la langue française

Les voyelles *orales* (i,e,u,...) sont émises sans intervention de la cavité nasale. Pour les voyelles *nasales*, le conduit nasal est couplé à la cavité buccale et l'émission se produit à la fois par les narines et par la bouche. Les consonnes

nasales font aussi intervenir la cavité nasale.
Les consonnes *fricatives* résultent de l'écoulement de l'air dans une *constriction* étroite en un point du conduit vocal, en particulier au niveau des lèvres et des dents. Les sons fricatifs sont non voisés (*f,s,...*) ou voisés (*v,ʒ,...*).
Les consonnes *occlusives* correspondent à des sons essentiellement dynamiques. Une forte pression est créée en amont d'une occlusion maintenue en un certain point du conduit vocal puis relachée brusquement. La période d'occlusion est appelée la phase de tenue. Pour les occlusives voisées un son basse fréquence est émis par vibration des cordes vocales pendant la phase de tenue; pour les occlusives non voisées, la tenue est un silence.

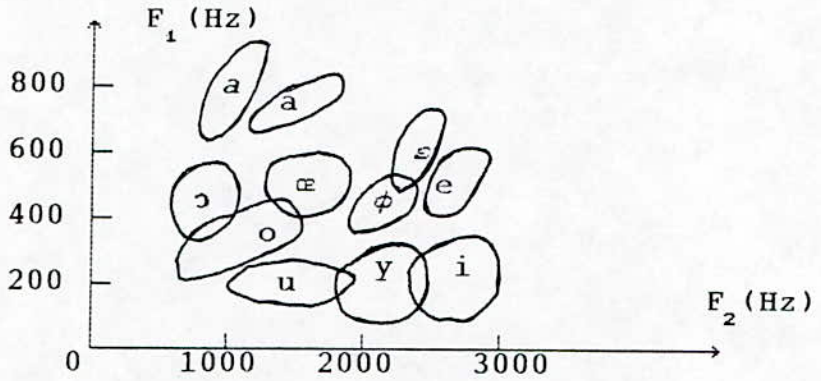


Fig-5 Représentation des voyelles dans le plan F_1-F_2

La figure 5 représente les voyelles dans le plan des deux premiers formants. On observe un certain recouvrement dans les zones de dispersion. Les trois premiers formants caractérisent déjà beaucoup mieux toutes les voyelles.
Pour les consonnes qui sont des sons brefs on observe plutôt des transitions formantiques.

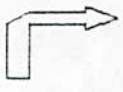
1.4-CONCLUSION

Le mécanisme de la production de la parole qui a été décrit dans ce chapitre conduit trés naturellement à une modélisation particulière appelée *modélisation autorégressive.*

Nous disposons là d'un outil trés commode pour procéder à l'analyse du signal vocal. On prendra quand même quelques précautions car l'hypothèse de stationnarité du signal n'est valable que durant de courts intervalles de temps.

Les procédures de modélisation doivent être adaptées pour en tenir compte.

Chapitre II



CHAPITRE 2 :

ANALYSE DE LA PAROLE

Au cours du chapitre précédent on a montré qu'un système linéaire tous-pôles peut modéliser la production des sons par le conduit vocal.

Maintenant,il nous semble utile de déterminer les coefficients a_i du modèle:

$$A(z) = \sum_{i=0}^{p} a_i . z^{-i}$$

p : ordre de la prédiction.

Tout d'abord,on admet les hypothèses qui justifient ce modèle.

La *stationnarité* du signal vocal garantie la validité du modèle sur des intervalles de l'ordre de 30ms. Il faut évidemment rafraichir les valeurs des coefficients a_i plusieurs fois par seconde.Toutefois, on voit bien qu'on a réalisé une compression de données trés efficace car, sur une durée de 30ms,pour une fréquence d'échantillonnage de 10KHz, on a 300 échantillons, et le *codage par prédiction linéaire* les ramène à près de 10 (si l'ordre de la LPC est 10).

2.1-RAPPEL : TRANSFORMEE DE FOURIER ET AUTOCORRELATION

Soit un signal X(t) sommable, on a:

$$X(\omega)=\int_{-\infty}^{+\infty} x(t).e^{-j\omega t}.dt \quad ; \quad X : \text{spectre.}$$

Dans le cas d'un signal périodique discret,on a:

$$X(k) = \sum_{i=0}^{N-1} x(i).\, e^{-jik/N}$$ C'est la DFT.

k : Fréquence discrète.

Pour les calculs numériques, on utilise de préférence la DFT. On connait un algorithme capable de calculer très éfficacement la DFT, c'est la FFT.
Celle-ci permet le calcul rapide de la transformée de Fourier sur un grand nombre de point.
En annexe de ce polycopié, on trouvera les programmes DFT et FFT.

On définit ainsi la fonction d'autocorrélation:

$$\emptyset(\tau)=\int_{-\infty}^{+\infty} x(t).x^{*}(t+\tau)\, dt$$

Propriété: $\emptyset(0) = \int_{-\infty}^{+\infty} (x(t))^2 \, dt$

On voit que $\emptyset(0)$ est l'énergie du signal.
D'autre part le spectre d'amplitude de la fonction d'autocorrélation est égal au spectre d'énergie du signal x.

2.2-ESTIMATION DU MODELE AUTOREGRESSIF

Soit x la réponse du système de transmittance $\frac{\sigma}{A(z)}$ à une excitation u:

$$X(z) = U(z) \cdot \frac{\sigma}{A(z)} \qquad (2\text{-}1)$$

U:excitation (*bruit blanc ou train périodique d'impulsion*).

σ :gain du modèle.

$$A(z) = \sum_{i=0}^{p} a(i) \cdot z^{-i} \quad , \quad a(0) = 1$$

Dans le domaine temporel celà aboutit à la récurrence suivante:

$$x(n) + \sum_{i=1}^{p} a(i).x(n-i) = \sigma.U(n) \qquad (2\text{-}2)$$

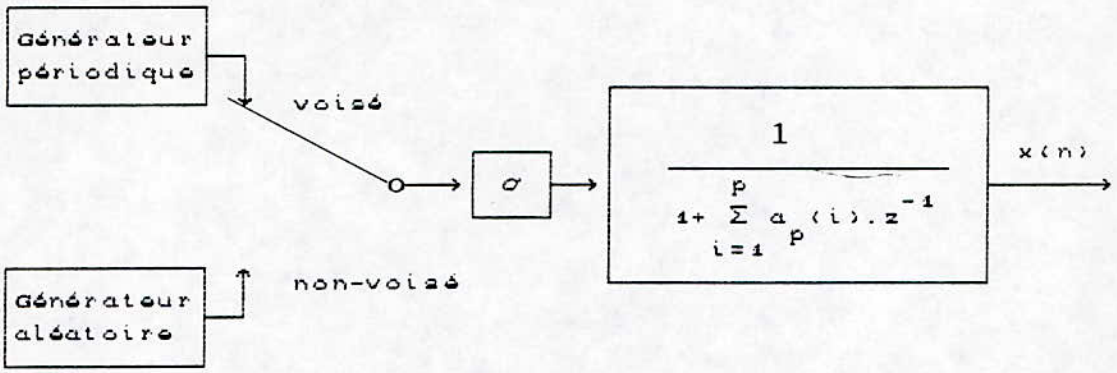


Fig 2-1 Modèle autorégressif de production de la parole

L'estimation des paramètres du modèle est basée sur l'observation du signal.

On définit de la manière suivante l'erreur de prédiction.

$$e(n) = \sum_{i=0}^{p} a(i)\ x(n-i) \qquad (2\text{-}3)$$

Le critère usuel pour estimer les paramètres est la *minimisation de la variance* (ou énergie) de l'erreur:

$$\sigma_e^2 = \Phi_{ee}(0) = \sum_{i=0}^{p} \sigma^2(i) = \sum_{x=0}^{p} (a(i) \cdot x(n-i))^2$$

$$\sigma_e^2 = \sum_{i=0}^{p} \sum_{j=0}^{p} a(i) \cdot a(j) \cdot x(n-i) \cdot x(n-j)$$

La fonction d'autocorrélation Φ_{xx} du signal vocal X apparait dans l'expression de σ_e^2 car $\Phi_{xx}(i-j) = x(n-i) \cdot x(n-j)$

Finalement on obtient la formule suivante:

$$\sigma_e^2 = \sum_{i=0}^{p} \sum_{j=0}^{p} a(i) \cdot a(j) \cdot \Phi_{xx}(i-j) \qquad 2\text{-}4$$

On veut minimiser σ_e^2 par rapport aux coefficients a(i):

$$\frac{d\,\sigma_e^2}{d\,a(i)} = \sum_{j=0}^{p} \Phi_{xx}(i-j) \cdot a(j) = 0 \quad ; \quad i = 1..p$$

Sachant que a(0)=1 on aboutit au système suivant:

$$\sum_{j=1}^{p} \Phi_{xx}(i-j) \cdot a(j) = -\Phi_{xx}(i) \quad ; \quad i = 1..p$$

C'est un système de *P* équations linéaires à *P* inconnues.

Utilisons la notation matricielle:

$a = [1, a(1), a(2), \ldots, a(p)]$

$\underline{a} = [a(1), a(2), \ldots, a(p)]$

La matrice d'autocorrélation $\Phi_{xx}^{(p)}$ prend cette forme:

$$\Phi_{xx}^{(p)} = \begin{bmatrix} \Phi_{xx}(0) & \Phi_{xx}(1) & \ldots & \Phi_{xx}(p) \\ \Phi_{xx}(1) & \ldots & & \vdots \\ \vdots & & & \vdots \\ \Phi_{xx}(p) & \Phi_{xx}(p-1) & \ldots & \Phi_{xx}(0) \end{bmatrix}$$

On peut faire deux observations.La première est que

$\Phi_{xx}^{(p)}$ est une matrice symétrique.

La seconde :les éléments de chaque diagonale parallèle à la principale sont égaux.

On peut donc affirmer que la matrice d'autocorrélation est une matrice de *Toeplitz symétrique.*

La variance de l'erreur de prédiction est une forme quadratique.

$\sigma_e^2 = a \cdot \Phi_{xx}^{(p)} \cdot a^T$ (a^T matrice transposée de a)

$\Phi_x = [\Phi_{xx}(1), \Phi_{xx}(2), \ldots \Phi_{xx}(p)]$

Le système d'équation 2-5 devient:

$$\Phi_{xx}^{(p-1)} \cdot \underline{a}^T = -\Phi_x^T \qquad (2\text{-}6)$$

Ou bien:

$$\begin{bmatrix} \Phi_{xx}(0) & \Phi_{xx}(1) & \dots & \Phi_{xx}(p-1) \\ \vdots & \vdots & \vdots & \vdots \\ \Phi_{xx}(p-1) & \dots & \dots & \Phi_{xx}(0) \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \\ \vdots \\ a(p) \end{bmatrix} = - \begin{bmatrix} \Phi_{xx}(1) \\ \Phi_{xx}(2) \\ \vdots \\ \Phi_{xx}(p) \end{bmatrix}$$

Pour obtenir les coefficients a(i), il suffit de résoudre ce système d'équation.On dispose pour celà d'un algorithme capable d'exploiter efficacement les propriétés des matrices de Toeplitz symétrique.C'est l'algorithme de *Levinson-Durbin.* Mais auparavant,il faut calculer les éléments de la matrice d'autocorrélation:

$$\Phi_{xx}(k) = \sum_{i=0}^{N-k-1} x(i).x(i+k)$$

2.2.1 -ESTIMATION DU GAIN DU MODELE:

Après avoir évalué les coefficients a(i),on veut choisir une valeur adéquate pour le gain du modèle:

$$\sigma = \sigma_{em} \qquad 2\text{-}7$$

σ^2_{em} :variance de l'erreur de prédiction après minimisation.

Or d'aprés l'équation 2-3 :

$$\sigma^2_{em} = \sum_{i=o}^{p} a(i).\phi_{xx}(i)$$

2.2.2-ALGORITHME DE LEVINSON-DURBIN

Rappelons que la fonction d'autocorrélation est supposée connue et que pour un signal stationnaire on a:

$$\emptyset_{xx}(i,j) = \emptyset_{xx}((i-j)) = \emptyset_{xx}(k)$$

Initialisation:

$$a_m(0) = 1 \ , \ (m = 1,2..p) \qquad \alpha_o = \emptyset_{xx}(0,0) = \sigma^2_x$$

Récursion:

-Pour $m = 1,2,...p$

$$k_m = -(1/\alpha_m \sum_{i=o}^{m-1} a_{m-1}(i).\emptyset_{xx}(m-i)$$

-Pour $i = 1,2,...m-1$

$$a_m(i) = a_{m-1}(i) + k_m.a_{m-1}(m-i)$$

$$a_m(m) = k_m$$

$$\alpha_m = \alpha_{m-1}(1-k^2_m)$$

2.2.3-COMMENTAIRE:

La méthode que nous avons décrite est la méthode de l'autocorrélation, elle offre un grand nombre d'avantages. La quantité de calcul n'est pas trop grande et surtout on peut garantir la stabilité du modèle AR et celle de l'algorithme de résolution.

2.3-ANALYSE DU SIGNAL VOCAL:

Avant toute chose,il faut fixer les conditions suivantes:
-La fréquence d'échantillonnage f_e.
-L'ordre de la prédiction P.
-Le nombre d'échantillon N par tranche d'analyse.
-Le decalage L d'une tranche à l'autre.
-La préaccentuation éventuelle.
-La fonction fenêtre W(n).

On choisit en général une fréquence d'échantillonnage f_e supérieur à 6Khz. Pour la reconnaissance de la parole une fréquence f_e de 10 KHz peut convenir.
L'ordre P de la prédiction se situe entre 8 et 16.

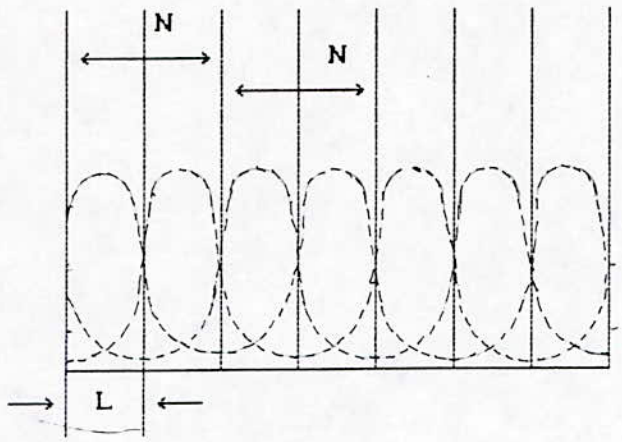


Fig 2-2 Découpage en tranche avec recouvrement

Pour la méthode de l'autocorrélation, l'erreur de prédiction e(n) prend parfois des valeurs exagérées, on est obligé, alors, de pondérer le signal avec une fonction fenêtre w(n). Celà nécessite un temps d'analyse suffisant.On utilise couramment des fenêtres de 30 ms décalées de 10 ms.
Le passage du signal dans un filtre de transmittance $(1-\mu.z^{-1})$ (μ compris entre 0,9 et 1), a pour effet d'accentuer la partie haute du spectre du spectre.Ce pré-traitement assure un bon conditionnement des algorithmes de résolution. Usuellement,on prend μ = 0,95.

La procédure à suivre pour analyser le signal vocal pour la méthode de l'autocorrelation est la suivante:

* aquisition du signal $s(n) = s(n\ T\ e)$

* préaccentuation par passage dans un filtre de transmittance $1-0,95.z^{-1}$

* Constitution de blocs de N échantillons avec décalages de L

échantillons.

* Application de la pondération par la fenêtre de Hamming:

$W(n)=0,54 - 0,4.\cos(2\pi.n/(N-1)) \qquad 0 \leq n \leq N-1$

$=0 \qquad$ ailleur

$x(n)= s(n).w(n)$

* Calcul de la matrice d'autocorrélation $R_x^{(p)}$:

$$r_x(k)= \sum_{n=0}^{n-1-k} x(n).x(n+k)$$

* résolution proprement dite du système. On obtient donc les coefficients de prédiction a(i).

2.4-ANALYSE HOMOMOPHIQUE

2.4.1-DEFINITION DU CEPSTRE:

On appelle *cepstre* du signal x la transformée en Z inverse du logarithme de la transformée du signal x.

On note le cepstre du signal par $\hat{x}$.

$$x(n) \xrightarrow{} (Z) \xrightarrow{X(z)} (Ln) \xrightarrow{Ln\ X(z)} (Z^{-1}) \xrightarrow{\hat{x}(n)}$$

2.4.2-Propriétés:

* si $x=x_1+x_2$ alors $\hat{x}=\hat{x}_1+\hat{x}_2$
* si *x est réel alors* $\hat{x}$ *est réel.*

2.4.3-CEPSTRE REEL:

Le cepstre complexe défini précédemment présente l'inconvénient majeur qu'il est difficile de reconstituer la phase du signal.
Mais les recherches effectuées sur l'audition ont largement prouvé que l'oreille est insensible à la phase d'un signal.
On en conclut que l'information sur la phase est inutile et qu'il est possible de définir le cepstre réel noté c(n):

$$C(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} Ln\ |x(e^{j\theta})|.e^{jn\theta}\ d\theta$$

On peut calculer le cepstre réel de deux manières.
La première utilise la FFT.
La seconde est présentée dans le paragraphe suivant.

2.4.4-RELATION DU CEPSTRE REEL AVEC LES COEFFICIENTS DE PREDICTION

Le calcul du cepstre réel par l'intermédiaire de la FFT est assez coûteux en temps de calcul. Il peut toutefois être estimé à partir des coefficients de prédiction $a_p(i)$:

On peut écrire:

$$\text{Ln}\ \frac{1}{Ap(\mathcal{Z})} = \sum_{n=1}^{\infty} C(n)\ \mathcal{Z}^{-n}$$

Le logarithme de la transmittance $1/A_p(\mathcal{Z})$ est bien égal à la transformée en Z du cepstre (d'aprés la définition)

On dérive chaque membre par rapport à $\mathcal{Z}^{-1}$:

$$\frac{A'p(\mathcal{Z})}{Ap(\mathcal{Z})} = \sum_{n=1}^{\infty} n.\ C(n).\ \mathcal{Z}^{-(n+1)}$$

Or $Ap(\mathcal{Z}) = \sum_{i=0}^{p} a_p(i).\ \mathcal{Z}^{-i}$ et $A'_p(\mathcal{Z}) = \sum_{i=1}^{p} i.\ a_p(i)\mathcal{Z}^{-i+1}$

$$-\sum_{i=0}^{p} i.\ a_p(i).\ \mathcal{Z}^{-i+1} = \left[\sum_{j=0}^{p} a_p(j).\ \mathcal{Z}^{-j}\right] \cdot \left[\sum_{n=1}^{\infty} n.\ C(n).\ \mathcal{Z}^{-n+1}\right]$$

Soit: $-i.\ a_p(i) = \sum_{n=1}^{i-1} n.\ C(n).\ a_p(i-n) + i.c(i)$

On obtient donc la récurrence suivante qui permet le calcul aisé du cepstre:

$$\boxed{c(i) = -a_p(i) - \sum_{n=1}^{i-1} (1-n/i).\ a_p(n).\ c(i-n)} \qquad i>0 \qquad (2\text{-}8)$$

Avec $c(0)=\ln \sigma^2$ qui contient l'information sur l'énergie du signal, σ :gain du modèle.

2.4.5-APPLICATION DU CEPSTRE REEL:

En principe, *l'analyse homomorphique* peut être utilisée pour isoler l'excitation u et la réponse impulsionnelle h ,du signal observé x:

$x = u+h \quad \Rightarrow \quad \hat{x} = \hat{u}+\hat{h}$

En pratique on se contente d'estimer la période du fondamentale et parfois l'enveloppe du spectre.

L'intérêt des coefficients cepstraux c(n), en reconnaissance de la parole est qu'ils sont independants de l'énergie du signal, en excluant C(0).

2.5-ESTIMATION DE LA TRAJECTOIRE DES FORMANTS

On a vu au chapitre 1 que les pôles du modèle autoregressif correspondent aux fréquences propres F_k du conduit vocal.

Les trois premiers formants sont essentiels pour caractériser le spectre vocal.

L'estimation des fréquences F_k et eventuellement des bandes passantes B_k est possible par une recherche des maximums du spectre du modèle. Les maximums du spectre du modèle peuvent être déterminés par la factorisation du polynôme $A_p(z)$.

A une racine $z_n = \rho_k . \exp(j.\theta)$ proche du cercle unité correspond un certain formant:

$$\hat{F}_k = \hat{\theta}_k / 2\pi . f_e$$

$$\hat{B}_k = (\Delta f)_{3dB} \simeq \frac{1}{\pi}(1-f_k).f_e \qquad (2\text{-}9)$$

Pour résoudre le polynôme $A_p(z)=0$ on utilisera l'algorithme de Bairstow.

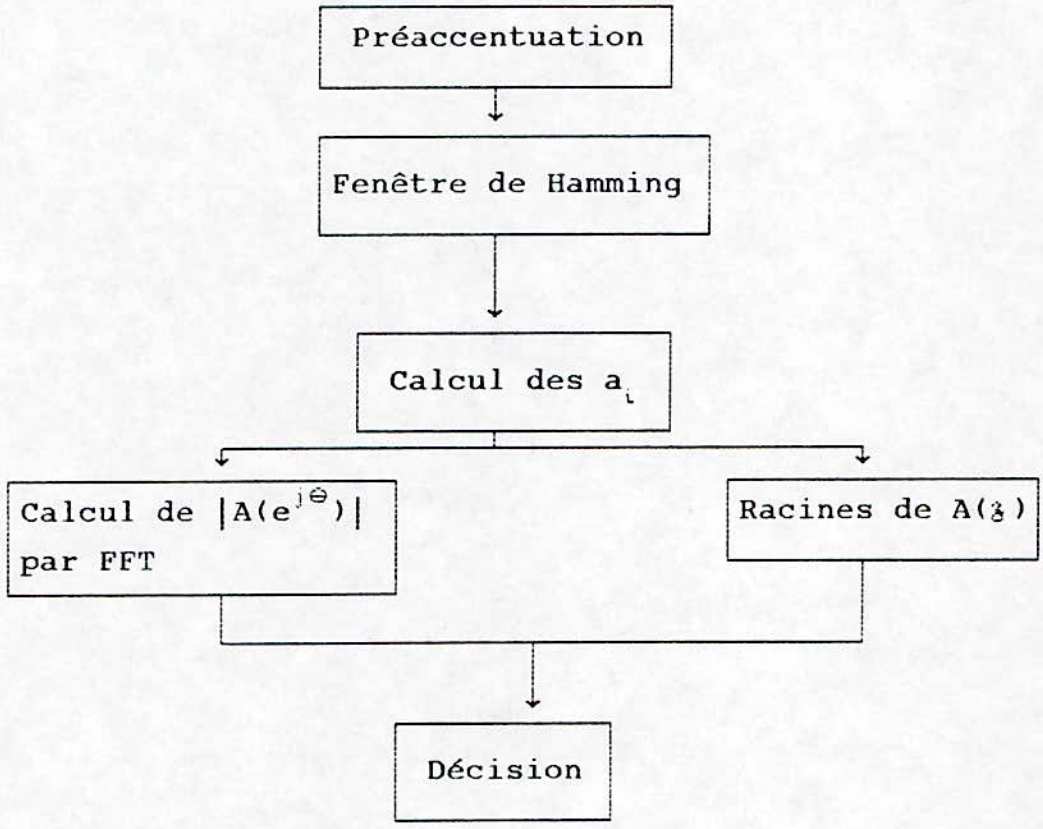


Fig2-3Estimation de la trajectoire des formants

Mais souvent on préfère localiser les maximums du spectre par calcul de $|A_p(e^{j\theta})|$ à l'aide de la FFT et puis on prend

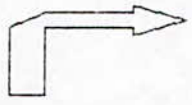
La localisation d'un maximum peut être améliorée par interpolation parabolique entre les valeurs de $1/|A_k|^2$.

Remarque

Il est en général difficile d'assurer de façon automatique la continuité des trajectoires des formants.Il est parfois impossible de distinguer deux formants s'ils sont trop proches.

Il est bon de savoir que les voyelles orales et les voyelles nasales correspondantes ne diffèrent que par les amplitudes relatives de leurs formants.

Chapitre III



CHAPITRE 3:

RECONNAISSANCE DE LA PAROLE

Il semble que l'oreille procède à une certaine analyse spectrale continue dont les résultats sont transmis au cerveau.L'identification d'un mot consiste donc en une succession de comparaisons de spectres,si l'on se limite au niveau acoustique.

Un spectre peut être représenté d'une façon objective par différents ensembles de paramètres;en particulier le spectre du modèle AR, qui correspond très bien à l'enveloppe du spectre vocal, peut être caractérisé par les coefficients $a_p(i)$.

Un ensemble ordonné de paramètres caractérisent un spectre vocal est appelé *vecteur acoustique* vecteur spectral.

Un mot est, bien sûr, constitué par une suite de vecteurs acoustiques.

La principale difficulté consiste à reconnaitre un mot. Pour celà on dispose de deux approches.La première exploite la notion de *distance* entre deux mots. L'algorithme DTW permet d'optimiser cette distance. Sur cette méthode,nous ne donnerons que quelques définitions.

Une approche apparement très différente consiste à spéculer sur un modèle statistique de production pour chaque mot du vocabulaire.

Nous allons approfondir et détailler cette méthode.

3.1-DISTANCES ET MESURES DE DISSEMBLANCES

3.1.1-DISTANCE EUCLIDIENNE:

Soit deux vecteurs acoustiques x et y de K composantes:

$$d(x,y)=\left(\sum_{i=1}^{k} (x_i - y_i)\right)^{1/2} \qquad (3\text{-}1)$$

3.1.2-DISTANCE CEPSTRALE:

La distance cepstrale est d'un usage très courant en reconnaissance de la parole.

On utilise les coefficients c(n) du cepstre réel. Ces coefficients sont calculés comme vu au chapitre précédent à partir des coefficients de prédiction linéaire:

La distance cepstrale est basée sur un nombre fini de termes:

$$d_{cEp} = (C_x(0) - C_y(0))^2 + 2.\sum_{l=1}^{L} (C_x(l) - C_y(l))^2 \qquad (3\text{-}2)$$

Le terme $(C_x(0) - C_y(0))^2$ qui correspond au gain du modèle ($C_x(0) = \ln \sigma_x^2$ et $C_y(0)=\ln \sigma_y^2$) peut ne pas être utilisé dans l'expression (3-2).

3.1.3-DISTANCE DE ITAKURA -SAITO:

Il s'agit en fait d'une *mesure de dissemblance* car la condition de symétrie n'est pas satisfaite:

$$d_{IS} = \frac{\delta_{xy}}{\alpha_y} - \frac{\text{Ln } \alpha_x}{\sigma_y^2} - 1 \qquad (3\text{-}3)$$

Avec: $\alpha_x = a_x . R_x . a_x^T$

$\delta_{xy} = a_y . R_x . a_y^T$

$\sigma_y^2 = \alpha_y^2 = a_y . R_y . a_y^T$

$a_x = 1, a_x(1), a_x(2), \ldots a_x(p)$ Vecteur des coefficients de la LPC pour X.

a_y : coefficients de la LPC pour y

R_x : matrice d'autocorrélation de x

R_y : // // de y

3.1.4-RAPPORT DE VRAISSEMBLANCE(LIKELIHOOD RATIO):

Cette mesure exploite les mêmes paramètres que la distance de Itakura-Saito,mais elle est beaucoup plus utilisée car elle prend moins de temps pour le calcul:

$$d_{LR}(x,y) = \frac{\delta_{xy}}{\alpha_x} - 1 = \frac{a_y . R_x . a_y^T}{a_x . R_x . a_x^T} - 1 \qquad (3\text{-}4)$$

3.2- METHODES STOCHASTIQUES

L'utilisation des méthodes statistiques ou stochastiques s'appuit sur l'hypothèse qu'un automate peut émettre un mot ou un phonème. A chaque etat interne de l'automate correspond l'émission d'un vecteur acoustique.

C'est donc la succession des états internes qui provoque la production d'un mot .On dit de ce modèle qu'il est doublement stochastique car d'une part les transitions d'un état à l'autre se font selon certaines probabilités et d'autre part l'émission d'un vecteur acoustique obéit à une probabilité qui dépend de l'état.

Ce modèle s'identifie tout naturellement à une chaîne de Markov qui est un processus causal.

En tout état de cause les variables de sortie X de notre modèle sont supposées observables. Les variables d'état ω ne le sont pas.

Comme nous le verrons, notre problème sera celui de déterminer à chaque instant l'état le plus probable compte tenue d'une séquence d'observation donnée et de la connaissance supposée des parametres du modèle.

Ce problème et le modèle qui lui est associé, justifient l'utilisation courante de l'expression anglo-saxonne de "Hidden Markov Modeling" (HMM).

Dans ce qui suit on détaillera et on développera toutes ces notions.

3.2.1-INTRODUCTION

3.2.1-1 CHAINES DE MARKOV D'ORDRE 1:

On considère un ensemble de L symboles.

Une chaîne de Markov d'ordre 1 est une séquence pour laquelle la probabilité d'occurence de chaque symbole ne dépend que de la nature du symbole précédent:

$$P(x_n/x_{n-1},x_{n-2},..x_1)=P(x_n/x_{n-1})$$

Si l'indice n s'identifie au temps ,on voit tout de suite qu'une chaîne de Markov est un processus causal.

D'après le théorème des probabilités totales:

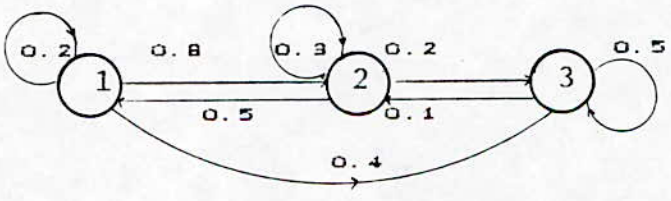
$$\sum_{i=1}^{L} P(x=\psi_i/x_{n-1}=\psi_j)=1$$

Une chaîne de Markov est représenté par une matrice de transition L*L dont les éléments sont des probabilités de transition d'un symbole à l'autre $P_{ij}=P(x_n=\psi_j/x_{n-1}=\psi_i)$

$$\text{avec } \sum_{j=1}^{L} P_{ij}=1$$

Exemple:

$$P=\begin{bmatrix}0.2 & 0.8 & 0\\ 0.5 & 0.3 & 0.2\\ 0.4 & 0.1 & 0.5\end{bmatrix} \qquad L=3$$



3.2.1-2 AUTOMATE FINI:

Définition: un automate fini est défini de la manière suivante:

-un alphabet d'entrée U, $u_t \in U$.

-un ensemble fini d'états S, $s_t \in S$.

-un alphabet de sortie X, $u_t \in X$.

u_t, s_t et x_t sont les valeurs à l'instant $t=1,2,...$

-une loi de transition vers l'état futur $s_{t+1} = \lambda(u_t, s_t)$.

-une loi de production de la sortie

$x_t = \delta(u_t, s_t)$ pour un automate de Mealy

$x_t = \delta(s_t)$ pour un automate de Moore

3.2.1-3 CARACTERISTIQUE DU MODELE UTILISE EN RECONNAISSANCE DE LA PAROLE

L 'automate choisi est autonome : il ne possède pas de variable d'entrée. La loi de transition vers l'état futur est une chaine de Markov. La loi de production de la sortie est une matrice stochastique. Chaque état émet un vecteur acoustique selon une loi de probabilité.

On a donc formulé le concept d'automate fini stochastique.

Exemple:

A: matrice de transition

$$A = \begin{bmatrix} 0.5 & 0.3 & 0.0 & 0.2 \\ 0.2 & 0.5 & 0.3 & 0.0 \\ 0.0 & 0.2 & 0.5 & 0.3 \\ 0.3 & 0.0 & 0.2 & 0.5 \end{bmatrix} \quad , \qquad \sum_j A_{ij} = 1$$

B: matrice d'émission ou de production

$$B = \begin{bmatrix} 0.9 & 0.05 & 0.05 \\ 0.1 & 0.8 & 0.1 \\ 0.0 & 0.1 & 0.9 \\ 0.5 & 0.5 & 0.0 \end{bmatrix} \quad , \qquad \sum_j B_{ij} = 1$$

Si $P(\omega_{\tau+1} = \psi_j / \omega_\tau = \psi_i)$ est la probabilité de transition d'un état i à un état j alors $P(X/\ \omega_\tau = \psi_i)$ est la probabilité d'émission du vecteur acoustique X par l'état i.

Si X est une variable discrète qui prend ses valeurs sur un ensemble de M élément, alors $\sum_k^M P(X_k / \omega_\tau = \psi_i) = 1$.

On voit bien que la matrice de transition est d'ordre L*L (L:nombre d'états) tandis que la matrice d'émission est d'ordre M * M.

Cette approche du problème de reconnnaissance d'un mot permet de prendre en considération toutes les possibilités de répétition et d'omission d'un son, qui sont observées dans les diverses prononciations d'un mot. Il semble judicieux de permettre avec certaines probabilités, l'émission de plusieurs sons par un même état ce qui entraine une réduction notable du nombre d'états des modèles.

3.2.2-DESCRIPTION DE LA DIFFICULTE DE L'EVALUATION DE LA VRAISEMBLANCE $\tilde{\mathcal{L}}$

Soit ω une variable aléatoire dont l'espace d'état discret est noté $\Psi=\{\psi_1,..\psi_o\}$. La notation $\omega^\tau=\psi_j$ indique que le processus est dans l'état ψ_j à l'instant τ.

La chaine de Markov est caractérisé par une distribution initiale $P_j=P(\omega^1=\psi_j)$, $j=1,..o$, et une matrice de transition $P_{jk}=P(\omega^{\tau+1}=\psi_k/\omega^\tau=\psi_j)$; $\tau>=1$.

Rappelons qu'une chaîne de Markov est définie de la manière suivante:

$$P(\omega^\tau/\omega^1,..\omega^{\tau-1})=P(\omega^\tau/\omega^{\tau-1}) \qquad (1)$$

d'autre part on suppose l'indépendance d'une séquence de vecteur acoustique etant donné la séquence d'états correspondante:

$$P(\bar{X}^T/\bar{\omega}^T)=P(X^1,..X^T/\omega^1..\omega^T)=\prod_{\tau=1}^{T} P(X^\tau/\omega^\tau) \qquad (2)$$

X^τ:variable aléatoire àl'instant τ (qu'on appelle aussi

observation).

La notation $\bar{X}_1^T = X^1, X^2, ..X^T$ désigne une séquence de variable aléatoire X^τ.

La notation $\bar{\omega}_1^T = \omega^1, \omega^2, ..\omega^T$ désigne une séquence de variable aléatoire ω^τ.

Notre objectif c'est de pouvoir calculer la vraisemblance $\tilde{\mathcal{L}}$ de la séquence $\bar{X}_1^T$ d'observation.

$$\tilde{\mathcal{L}} = P(\bar{X}_1^T) = P(X_1, X_2, ..X^T)$$

Appliquons le théorème de la probabilité totale:

$$\tilde{\mathcal{L}} = P(\bar{X}_1^T) = \sum_{\omega^1 ..\omega^T = \Psi_1}^{\Psi} P(\bar{\omega}_1^T).P(\bar{X}_1^T/\bar{\omega}_1^T)$$

Or d'après (2) : $P(\bar{X}_1^T/\bar{\omega}_1^T) = \prod_{\tau=1}^{T} P(X^\tau/\omega^\tau)$

Et d'après le théorème des probabilités composées:

$$P(\bar{\omega}_1^T) = P(\omega^1).\prod_{\tau=2}^{T} P(\omega^\tau/\omega^{\tau-1})$$

Finalement:

$$\tilde{\mathcal{L}} = P(\bar{X}_1^T) = \sum_{\omega^1 ..\omega^T = \Psi_1}^{\Psi} P(\omega^1).P(X^1/\omega^1).\prod_{\tau=2}^{T} P(\omega^\tau/\omega^{\tau-1}).P(X^\tau/\omega^\tau) \qquad (3)$$

En réalité cette dernière expression est totalement inutilisable car on sera confronté à un calcul énorme de l'ordre de $2.T.\Theta^T$ (T:longueur ou durée de la séquence; Θ: nombre d'états).

Par bonheur la causalité nous permettra de linéariser la complexité du calcul. pour celà on dispose de deux méthodes.
La première calcule la vraisemblance totale, on dispose de l'algorithme de Baum-welch.
La second cherche la probabilité maximum,l'algorithme de Viterbi fait ce calcul.

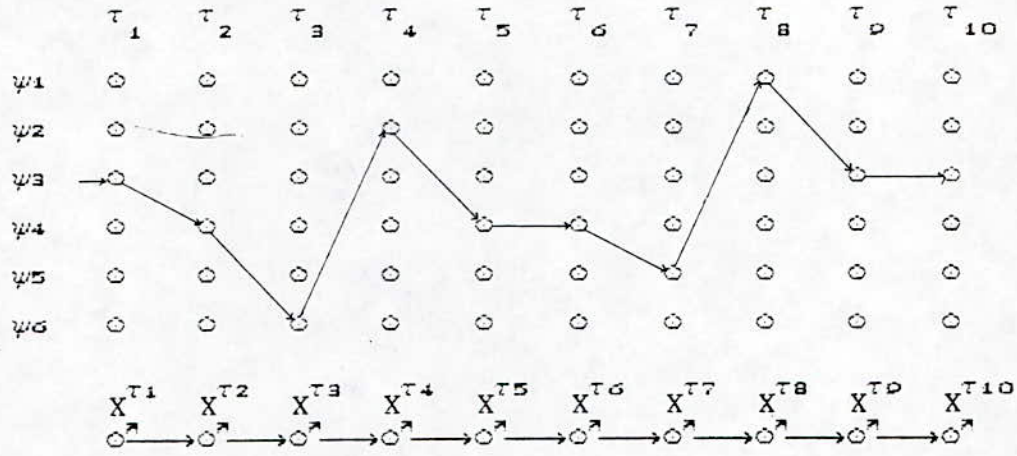


Fig-1 Représentation en graphe d'un modèle doublement stochastique

Donnons l'interprétation physique de ce modèle abstrait: les observations X^{τ} peuvent être un vecteur de coefficients de prédiction linéaire ou des coefficients cepstraux, les états ω^{τ} peuvent servir à caractériser la classe phonétique d'un segment de parole.
Avant de terminer ce paragraphe, on donnera quelques propriétés utiles des chaines de Markov.Il est bien connu que conditionnellement à ω^{τ}, les séquences $\{\omega^{t}\}_{1}^{\tau-1}$ et $\{\omega^{t}\}_{\tau+1}^{T}$ sont mutuellement indépendantes.De celà il est aisé de déduire les formules ci-dessous:

$$P(X^{\tau+1}/\omega^{\tau}=\psi_j, \bar{X}_1^{\tau}) = P(X^{\tau+1}/\omega^{\tau}=\psi_j) \qquad (4)$$

$$P(\bar{X}_{\tau+1}^{T} / \omega^{\tau}=\psi_j, X^{\tau}) = P(\bar{X}_{\tau+1}^{T} / \omega^{\tau}=\psi_j) \qquad (5)$$

D'une manière générale on a:

$$P(\bar{X}_1^{\tau}, \bar{X}_{\tau+1}^{T} / \omega^{\tau}=\psi_j) = P(\bar{X}_1^{\tau} / \omega^{\tau}=\psi_j) . P(\bar{X}_{\tau+1}^{T} / \omega^{\tau}=\psi_j) \qquad (6)$$

3.2.3-ALGORITHMES DE RECONNAISSANCE

La question qui va nous occuper présentement est celle d'obtenir des algorithmes efficaces permettant de calculer la probabilité des états, étant donné les observations et les paramètres du modèle sous-jacent.

3.2.3-1 RECONNAISSANCE PAR EVALUATION DE LA PROBABILITE TOTALE (BAUM-WELCH)

On veut évaluer la probabilité conjointe (ou vraisemblance) $\tilde{\mathcal{E}}_{\tau}(\psi_j)=P(\omega^{\tau}=\psi_j, \bar{X}_1^{T})$ de l'état ψ_j au temps τ et de la séquence d'observation $\bar{X}_1^{T}$.

Nous abordons le problème en remarquant que nous disposons de la décomposition suivante:

$$\tilde{\mathcal{E}}_{\tau}(\psi_j) = P(\omega^{\tau}= \psi_j, \bar{X}_1^{T}) = P(\omega^{\tau}=\psi_j, \bar{X}_1^{\tau}, \bar{X}_{\tau+1}^{T})$$

D'après le théorème des probabilités composées:

$$\tilde{\mathcal{E}}_{\tau}(\psi_j) = P(\omega^{\tau}= \psi_j, \bar{X}_1^{\tau}) \cdot P(\bar{X}_{\tau+1}^{T} / \omega^{\tau}=\psi_j, \bar{X}_1^{\tau})$$

D'après la relation (4):

$$P(\bar{X}^{\tau+1}/\omega^{\tau}=\psi_j, \bar{X}_1^{\tau}) = P(\bar{X}^{\tau+1}/\omega^{\tau}=\psi_j)$$

Finalement:

$$\tilde{\mathcal{E}}_{\tau}(\psi_j) = P(\omega^{\tau}=\psi_j, \bar{X}_1^{\tau}) \cdot P(\bar{X}_{\tau+1}^{T}/\omega^{\tau}=\psi_j) \qquad (7)$$

On note $\tilde{\mathcal{F}}_{\tau}(\psi_j) = P(\omega^{\tau}=\psi_j, \bar{X}_1^{\tau})$ la probabilité forward

et $\tilde{\mathcal{E}}_{\tau}(\psi_j) = P(\bar{X}_{\tau+1}^{T}/\omega^{\tau}=\psi_j)$ la probabilité backward

Ces deux probabilités forward et backward peuvent être calculées respectivement par une récurrence progressive et régressive. Nous admettons alors que la probabilité $\tilde{\mathcal{F}}_{\tau}(\psi_j)$ peut se calculer par la récurrence suivante :

$$\tilde{\mathcal{F}}_{\tau}(\psi_j) = P_j(X^{\tau}) \sum_{i}^{\theta} \tilde{\mathcal{F}}_{\tau-1}(\psi_j) . P_{ij} \qquad (8)$$

Pour $\tau=1$ $\quad \tilde{\mathcal{F}}_{\tau}(\psi_j) = P(\omega^1=\psi_j) . P(X^1/\omega^1=\psi_j) = P_j . P_j(X^1)$

Nous pouvons décomposer la probabilité backward $\tilde{\mathcal{E}}_{\tau}(\psi_j)$ de la manière suivante:

$$\tilde{\mathcal{E}}_{\tau}(\psi_j) = \sum_{k}^{\theta} \tilde{\mathcal{E}}_{\tau+1}(\psi_k) . P_{jk} . P_k(X^{\tau+1}) \qquad \begin{matrix}(9)\\ \tau=T-1,..1\end{matrix}$$

Avec $P_{jk}=P(\omega^{\tau+1}=\psi_k/\omega^{\tau}=\psi_j)$

et $P_k(X^{\tau+1}) = P(X^{\tau+1}/\omega^{\tau+1}=\psi_k)$

Si le modèle ne possède pas d'état terminal :

$\tilde{\mathcal{B}}_\tau(\psi_j)=1$ pour $\tau=\tau$

Les récurrences (8) et (9) permettent de calculer de manière efficace $\tilde{\mathcal{F}}_\tau(\psi_j)$ et $\tilde{\mathcal{B}}_\tau(\psi_j)$.

Les probabilités $\tilde{\mathcal{F}}_\tau(\psi_j)$ sont calculées par (8) et mémorisées au cours d'une passe progressive. Ensuite au cours d'une passe régressive les probabilités $\tilde{\mathcal{B}}_\tau$ sont calculées par (9). Et le produit (7) peut être effectué, ce qui nous donne les probabilités attendues $\tilde{\mathcal{E}}_\tau(\psi_j)$ pour $j=1..\mathcal{Q}$ et $\tau=T,..1$.

On remarque que :

$$\tilde{\mathcal{E}}_\tau(\psi_j)=P(\omega^\tau=\psi_j,\overline{X}_1^T) \Rightarrow \sum_j P(\omega^\tau=\psi_j,\overline{X}_1^T)=\tilde{\mathcal{E}}$$

$\tilde{\mathcal{E}}=P(X^1,..X^T)$ uniformément en τ

On peut observer que l'expression $\tilde{\mathcal{E}}=P(X^1,..X^T)=P(\overline{X}_1^T)$ rencontrée précédemment peut aussi s'ecrire, uniformément en τ, :

$$\tilde{\mathcal{E}}=\sum_j \tilde{\mathcal{E}}_\tau(\psi_j) = \sum_j \tilde{\mathcal{F}}_\tau(\psi_j).\tilde{\mathcal{B}}_\tau(\psi_j). \qquad (10)$$

Il est donc évident que les deux récurrences (8) et (9) nous ont permis de nous affranchir de la complexité exponentielle qui caractérise $\tilde{\mathcal{E}}$ en (3).

La représentation en graphe fournit à nouveau une interprétation intuitive évidente de l'algorithme forward-backward. Ainsi , par exemple, la probabilité $\tilde{\mathcal{F}}_{\tau 5}(\psi_4)$ est-elle la somme des probabilités d'émission de la sous-séquence $X^{\tau 1}..X^{\tau 2}$ le long de tous les chemins distincts menant au noeud $\omega^{\tau 5}=\psi_4$.

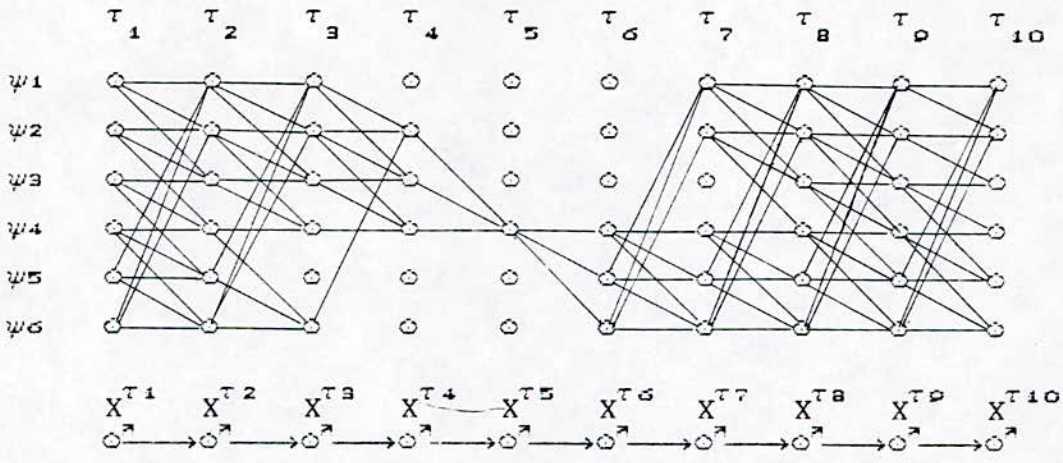


Fig-2: $\tilde{\mathcal{F}}_{\tau 5}(\psi_4)$. $\tilde{\mathcal{B}}_{\tau 5}(\psi_4)$ *est la somme des probabilités d'émission de la sous-séquence* $X_{\tau 1}..X_{\tau 5}$ *le long de tous les chemins distincts passant par le noeud* $\omega^{\tau 5}=\psi_4$

La probabilité backward, ainsi que la vraisemblance $\tilde{\mathcal{L}}_\tau(\psi_j)$ peuvent être interprétées de la même manière. Toutefois, on doit signaler que la mise en oeuvre des récurrences dans un programme d'ordinateur donnerait inévitablement lieu à des problèmes de dépassements numériques inférieurs de capacité (valeurs trops petites). Ceci s'explique par le fait que $\tilde{\mathcal{F}}_\tau(\psi_j)$ et $\tilde{\mathcal{B}}_\tau(\psi_j)$ tendent vers zéro de manière exponentielle. Diverses solutions à cette difficulté ont été formulées. Devijver a reformulé le calcul en terme de probabilité à postériori.

Nous présentont brièvement cette solution.

Soit $\mathcal{L}_\tau(\psi_j) = P(\omega^\tau=\psi_j / \bar{X}_1^T)$

$$= \frac{P(\omega^\tau=\psi_j, \bar{X}_1^T)}{P(\bar{X}_1^T)} \cdot \frac{P(\bar{X}_{\tau+1}^T / \omega^\tau=\psi_j)}{P(\bar{X}_{1+\tau}^T / \bar{X}_1^\tau)}$$

$$= \mathcal{F}_\tau(\psi_j) \cdot \mathcal{B}_\tau(\psi_j) \qquad , \; j=1..\theta$$

avec :

$$\mathcal{F}_\tau(\psi_j) = \frac{P(\omega^\tau = \psi_j, \bar{X}_1^T)}{P(\bar{X}_1^T)} \quad \text{et} \quad \mathcal{B}_\tau(\psi_j) = \frac{P(\bar{X}_{\tau+1}^T / \omega^\tau = \psi_j)}{P(\bar{X}_{1+\tau}^T / \bar{X}_1^\tau)}$$

La probabilité $\mathcal{F}_\tau(\psi_j)$ peut être calculée à l'aide d'une récurrence progressive simple:

$$\begin{aligned} \mathcal{F}_\tau(\psi_j) &= N_\tau . P_j . P_j(X^1) && ,\tau = 1 \\ &= N_\tau . \sum_i \mathcal{F}_{\tau-1}(\psi_i) . P_{ij} . P_j(X^\tau) && ,\tau = 2,..T \end{aligned} \qquad (11)$$

N_τ est un facteur de normalisation :

$$N_\tau = \left[\sum_j P_j . P_j(X^\tau) \right]^{-1} \qquad , \tau = 1$$

$$= \left[\sum_i \sum_j \mathcal{F}_{\tau-1}(\psi_i) . P_{ij} . P_j(X^\tau) \right]^{-1} \qquad ,\tau = 2,..T$$

La probabilité backward donne la récurrence régressive suivant:

$$\begin{aligned} \mathcal{B}_\tau(\psi_j) &= 1 && ,\tau = T \\ &= N_{\tau+1} . \sum_k \mathcal{B}_{\tau+1}(\psi_k) . P_{ik} . P_k(X^{\tau+1}) && ,\tau = T-1,..1 \end{aligned} \qquad (12)$$

En vertu de la définition de $\mathcal{E}_\tau(\psi_j)$ il vient :

$$\sum_j \mathcal{E}_\tau(\psi_j) = \sum_j \mathcal{F}_\tau(\psi_j) . \mathcal{B}_\tau(\psi_j) = 1 \qquad (13)$$

En reconnaissance l'évaluation de l'une des deux probabilités forward ou backward peut suffire car elles donnent qualitativement la même information que la vraisemblance $\tilde{\mathcal{E}}$. Parfois, on cherche à connaitre l'état du système au temps τ. aussi un estimateur $\hat{\omega}^\tau$ de l'état du système au temps τ s'obtient par la règle du maximum de probabilité à postériori

$$\hat{\omega}^\tau = \psi_j \text{ si } \tilde{\mathcal{E}}_\tau(\psi_j) = \max_i \tilde{\mathcal{E}}_\tau(\psi_i)$$

3.2.3-2 ALGORITHME DE VITERBI OU CALCUL DE LA PROBABILITE MAXIMUM

La vraisemblance $\tilde{\mathcal{L}}$ qu'on a déjà vu, se calcule selon la formule suivante :

$$\tilde{\mathcal{L}} = P(\bar{X}_1^T) = \sum_{\omega^1 \ldots \omega^T = \psi_i}^{\Psi} P(\omega^1).P(X^1/\omega^1).\prod_{\tau=2}^{T} P(\omega^\tau/\omega^{\tau-1}).P(X^\tau/\omega^\tau)$$

On a déjà précisé que ce calcul est fastidieux. toutefois Viterbi suggère de ne chercher que la probabilité maximum au lieu de la probabilité totale $\tilde{\mathcal{L}}$:

$$P_{max}(X) = P(\omega^1).P(X^1/\omega^1).\prod_{\tau=2}^{T} P(\omega^\tau/\omega^{\tau-1}).P(X^\tau/\omega^\tau)$$

On doit trouver une succession d'états ω^τ qui donnent la probabilité maximum P_{max}

L'algorithme de Viterbi sert à déterminer le parcours optimum dans un automate stochastique, c'est-à-dire celui le long duquel la probabilité de produire un mot donnée est maximum. Un certain nombre de parcours possibles aboutissent à l'état ψ_j à l'instant τ , on ne retient que ceux auquels est associée la plus grande probabilité d'avoir émis la séquence $\bar{X}_1^\tau$; ces parcours sont dits survivants à l'instant τ .Soit $P_\tau(\psi_i)$ cette probabilité maximum, on peut écrire:

$$P_\tau(\psi_i) = \underset{i}{Max} \left[P_\tau(\psi_j).\ P(\omega^\tau = \psi_j/\omega^{\tau-1} = \psi_i) \right] .P(X^\tau/\omega^\tau = \psi_j) \qquad (14)$$

L 'état ψ_i qui rend maximum l'expression entre crochet est le prédecesseur de l'état ψ_j dans le parcours optimal.

Si celà est necessaire, un retour en arrière le long du parcours optimal permet d'identifier la chaîne de Markov des états qui ont produit le mot.

Pour les calculs numériques la relation (14) est remplacée par une somme de logarithme: (15)

$$-\log P_{\tau}(\psi_i) = -\log P(X^{\tau}/\omega^{\tau}=\psi_j) - \min_i \left[\log P_{\tau}(\psi_j) . P(\omega^{\tau}=\psi_j/\omega^{\tau-1}=\psi_i) \right]$$

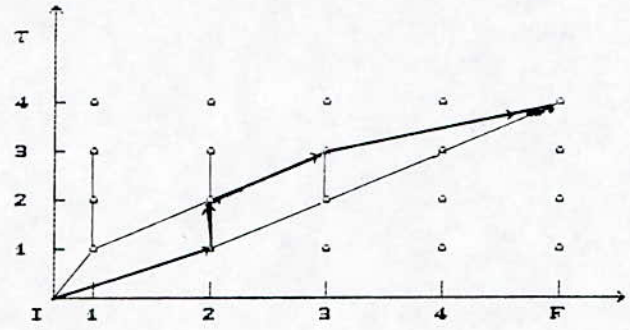


fig-3 Chemin optimal dans l'algorithme de Viterbi

3.2.4-ALGORITHME D'APPRENTISSAGE

On désire entraîner le modèle M associé à un mot X à partir de différentes versions de ce mot.

Il faut en premier lieu choisir le nombre Θ d'états émetteurs et fixer les valeurs initiales pour toutes les probabilités d'émission et de transition ($P(X^{\tau}/\omega^{\tau})$ et $P(\omega^{\tau+1}/\omega^{\tau})$).

L'entrainement d'un modèle consiste à modifier ses paramètres afin d'optimiser la reconnaissance des différentes versions du mot X.

La vraisemblance $\tilde{\varepsilon}$ d'une séquence $\bar{X}_1^T = \{X^1, .. X^T\}$ d'observation s'écrit comme on l'a déjà vu à la section 2:

$$\tilde{\mathcal{L}} = P(\bar{X}_1^T) = \sum_j \tilde{\mathcal{F}}_\tau(\psi_j).\tilde{\mathcal{B}}_\tau(\psi_j)$$

$$= \sum_{i=1} \sum_{j=1} \tilde{\mathcal{F}}_\tau(\psi_i).P_{ij}.P_j(X^{\tau+1}).\tilde{\mathcal{B}}_{\tau+1}(\psi_j) \quad ,(16)$$

En supposant donnés le nombre d'états θ et la séquence d'observation $\bar{X}_1^T$, nous allons considérer $\tilde{\mathcal{L}}$ comme une fonction des paramètres $\{P_i, P_{ij}, P_j(x_i)\}$ et formuler notre problème d'apprentissage comme celui de l'estimation des paramètres qui maximisent la vraisemblance $\tilde{\mathcal{L}}$ de $\bar{X}_1^T$, sous les contraintes $\sum_i P_i = 1$, $\sum_j P_{ij} = 1 \ \forall i$ et $\sum_i P_j(x_i) = 1 \ \forall j$.

Notre problème d'optimisation se prète bien à l'utilisation de la technique des multiplicateur de Lagrange. Nous formons dès lors la fonction auxilliaire suivante:

$$\boxed{\Pi = \tilde{\mathcal{L}} + \alpha.\left[\sum_i P_i - 1\right] + \sum_i \beta_i.\left[\sum_j P_{ij} - 1\right] + \sum_i \gamma_i.\left[\sum_k P_i(x_k) - 1\right]} \quad (17)$$

Où α, β_i et γ_i $(i,l=1..\theta)$ sont les multiplicateurs de Lagrange.

L'équation (17) montre que l'optimisation par rapport à une famille de paramètres peut être effectuée indépendemment de l'optimisation par rapport aux autres familles de paramètres. On se bornera à traiter le problème de l'optimisation par rapport aux probabilités de transition P_{ij}. En égalant à zéro la dérivée partielle de Π par rapport à P_{ij}, nous obtenons ceci:

$$\frac{\partial \Pi}{\partial P_{ij}} = \frac{\partial \tilde{\mathcal{L}}}{\partial P_{ij}} + \beta_i = 0 \qquad (18)$$

Une multiplication par P_{ij} nous fournit ce qui suit:

$$P_{ij}.\partial\tilde{\mathcal{L}}/\partial P_{ij}+\beta_i.P_{ij}=0 \qquad (19)$$

En vertu de la contrainte $\sum_i P_{ij}=1$, on obtient:

$$\beta_i=-\sum_j P_{ij}.\partial\tilde{\mathcal{L}}/\partial P_{ij} \qquad (20)$$

En substituant (20) dans (19) on obtient la solution sous la forme suivante:

$$P_{ij}=\frac{P_{ij}.\partial\tilde{\mathcal{L}}/\partial P_{ij}}{\sum_j P_{ij}.\partial\tilde{\mathcal{L}}/\partial P_{ij}} \qquad (21)$$

Examinons à présent le problème de l'évaluation de la dérivée partielle $\partial\tilde{\mathcal{L}}/\partial P_{ij}$.
En premier lieu, nous exprimons $\tilde{\mathcal{L}}$ comme la somme des vraisemblances de $\bar{X}_1^T$ le long de tous les chemins possibles distincts dans la représentation en graphe du modèle doublement stochastique .

$$\tilde{\mathcal{L}}=\sum_{\tau=1}^{T-1}\sum_{k=1}\sum_{l=1} P(\bar{X}_1^T,\omega^\tau=\psi_k,\omega^{\tau+1}=\psi_l) \qquad ,(22)$$

Nous remplaçons chaque terme de la triple somme par sa décomposition forward-backward. Ce faisant nous obtenons:

$$\tilde{\mathcal{L}}=\sum_{\tau=1}^{T-1}\sum_{i=1}\sum_{j=1}\tilde{\mathcal{F}}_\tau(\psi_i).P_{ij}.P_j(X^{\tau+1}).\tilde{\mathcal{B}}_{\tau+1}(\psi_j) \qquad ,(23)$$

$$\partial\tilde{\mathcal{L}}/\partial P_{ij}=\sum_{\tau=1}^{T-1}\tilde{\mathcal{F}}_\tau(\psi_i).P_j(X^{\tau+1}).\tilde{\mathcal{B}}_{\tau+1}(\psi_j) \qquad ,(24)$$

$$P_{ij} \cdot \partial \tilde{\mathcal{L}} / \partial P_{ij} = \sum_{\tau=1}^{T-1} \tilde{\mathcal{F}}_\tau(\psi_i) \cdot P_{ij} \cdot P_j(X^{\tau+1}) \cdot \tilde{\mathcal{B}}_{\tau+1}(\psi_j) \quad ,(25)$$

En substituant ces valeurs en (21), on obtient la formule de ré-estimation de P_{ij} :

$$P_{ij} = \frac{\sum_{\tau=1}^{T-1} \tilde{\mathcal{F}}_\tau(\psi_i) \cdot P_{ij} \cdot P_j(X^{\tau+1}) \cdot \tilde{\mathcal{B}}_{\tau+1}(\psi_j)}{\sum_{\tau} \tilde{\mathcal{F}}_\tau(\psi_i) \cdot \tilde{\mathcal{B}}_\tau(\psi_i)} \quad (26)$$

$$\tilde{\mathcal{L}}_\tau(\psi_i) = \sum_{\tau} \tilde{\mathcal{F}}_\tau(\psi_j) \cdot \tilde{\mathcal{B}}_\tau(\psi_j)$$

Pour P_i on a:

$$P_i = \frac{\tilde{\mathcal{F}}_1(\psi_i) \cdot \tilde{\mathcal{B}}_1(\psi_i)}{\sum_j \tilde{\mathcal{F}}_1(\psi_j) \cdot \tilde{\mathcal{B}}_1(\psi_j)} \quad ,(27)$$

Si on adopte une loi d'émission gaussienne pour des variables indépendantes :

$$\hat{\mu}_j[l] = \frac{\sum_{\tau=1}^{T-1} \tilde{\mathcal{F}}_\tau(\psi_j) \cdot X^\tau[l] \cdot \tilde{\mathcal{B}}_\tau(\psi_j)}{\sum_{\tau} \tilde{\mathcal{F}}_\tau(\psi_j) \cdot \tilde{\mathcal{B}}_\tau(\psi_j)} \quad ,(28)$$

$$\hat{\sigma}_j[l] = \frac{\sum_{\tau=1}^{T-1} \tilde{\mathcal{F}}_\tau(\psi_j) \cdot \tilde{\mathcal{B}}_\tau(\psi_j) \cdot (X^\tau[l] - \mu_j[l])}{\sum_{\tau} \tilde{\mathcal{F}}_\tau(\psi_j) \cdot \tilde{\mathcal{B}}_\tau(\psi_j)} \quad ,(29)$$

$X^\tau[l]$: composante de rang l du vecteur X^τ.

La moyenne $\hat{\mu}_j$ et la variance $\hat{\sigma}_j^2$ caractérisent la probabilité d'émission $P_j(X^T)$.Les formules (26)-(29) nous fournissent le principe de la méthode itérative de ré-estimation des paramètres P_i,P_{ij} et $P_j(X^T)$.
Cette procédure doit être reprise pour une nouvelle itération à partir de toutes les versions de X jusqu'à ce que les paramètres du modèle M se soient stabilisés.
L'algorithme que nous avons décrit est convergeant, toutefois il peut très bien converger vers un optimum local au lieu d'un optimum global.

3.2.5-MODELE DE BAKIS

Pour simplifier les modèles et réduire les calculs, on utilise avantageusement les modèles de Bakis. En fait dans ces modèles on admet que certaines probabilités de transition sont nulles.

3.2.5-1 MODELE DE BAKIS D'UN MOT ISOLE:

Pour un mot isolé le modèle de Bakis comporte N états émetteurs, un état I initial et un état final F.
Les états initial I et final F n'émettent aucun symbole.
Les probabilités de transition a_{ij} (qu'on notait précédemment P_{ij}) répondent aux conditions suivantes:
$a_{ij}=0$ Si $j<i$ ou $j>i+2$

D'autre part $a_{II}=0$ et $a_{FF}=1$

La figure ci-dessous illustre bien ce modèle.

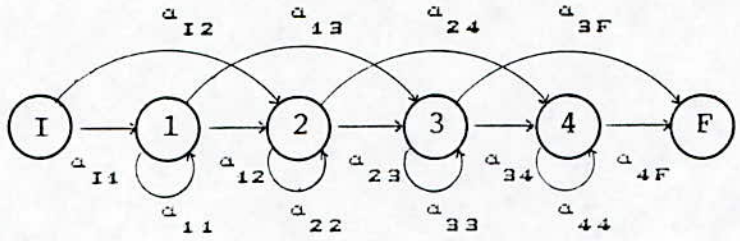


fig-4 Modèle de Bakis pour 4 états émetteurs

On rappelle que le nombre d'états dans un modèle dépend beaucoup du nombre de phonème dans le mot correspondant.

3.2.5-2 MODELE DE BAKIS POUR LES PHONEMES:

Pour modéliser un phonème on peut utiliser un modèle fixe comportant trois états émetteurs.

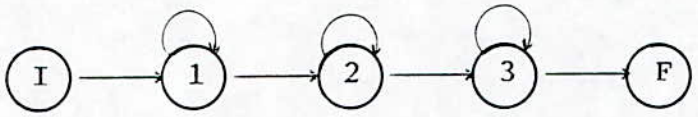


Fig-5 Modèle de phonème

Les sauts sont interdits: l'expérience montre que leur probabilité reste faible si l'admet à priori. L'état 2 représente la partie stable, et les états 1 et 3 les zones de transitions du phonème.
En pratique ce modèle fixe n'est pas utilisé tel quel puisqu'un phonème est rarement isolé.

3.2.5-3 MODELE DE BAKIS-JELINEK POUR DES MOTS ENCHAINES

Chaquz mot du vocabulaire est représenté par son modèle de

Bakis . La jonction des modèles est obtenue par une transition directe entre le dernier état d'un mot et le premier état du mot suivant:

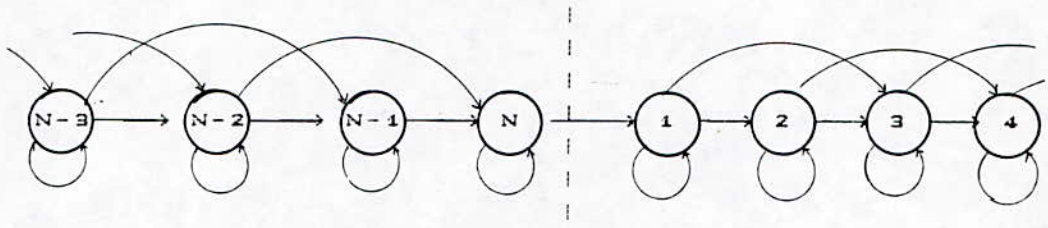


Fig-6 Modèle de Bakis-Jelinek pour des mots enchaînés

Remarque :les algorithmes de reconnaissance et d'apprentissage sont bien sûr toujours applicables avec des modèles de Bakis, mais ils doivent tenir compte des probabilités a_{iF} (ou P_{iF}) de transition de l'état i vers l'état final .

3.2.6-RECONNAISSANCE PHONEMIQUE

Il semble naturel d'espérer qu'une méthode efficace d'identification directe des phonèmes dont le nombre est très limité dans chaque langue, permettrait la reconnaissance de grands vocabulaires: mots isolés, phrases courtes et à la limite parole continue.
On peut envisager la reconnaissance de mots isolés ou même de courtes phrases considérés comme une suite de phonèmes enchaînés.On dispose pour celà du modèle fixe de Bakis pour les phonèmes. La jonction entre phonème serait organisée

comme le modèle de Bakis-Jelinek pour mots enchainés.
De cette manière on oublie la co-articulation qui sera prise en compte par l'entrainement qui est effectué necessairement sur des chaînes de phonèmes.

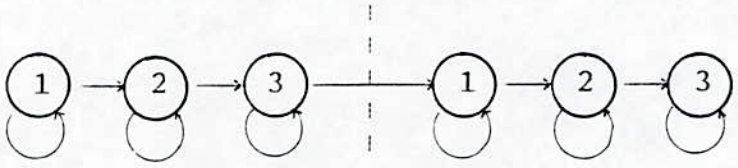


Fig-7 Modèle de phonèmes enchainés

La reconnaissance peut être organisée au niveau du mot; on construit un modèle pour chaque mot du vocabulaire par concaténation de modèles de phonèmes. On est très vite limité dans cette voie pour les vocabulaires importants.
On peut aussi procéder en deux étapes; on effectue en premier lieu un étiquetage des phonèmes par reconnaissance classique de phonèmes enchainés: algorithme de Viterbi suivi d'un retour en arrière dans le chemin optimal. En second stade, on valide la reconnaissance d'un mot en comparant la suite des phonèmes obtenus à la transcription phonétique de chaque mot du vocabulaire .
Le volume de calcul impliqué par une telle méthode reste très important.

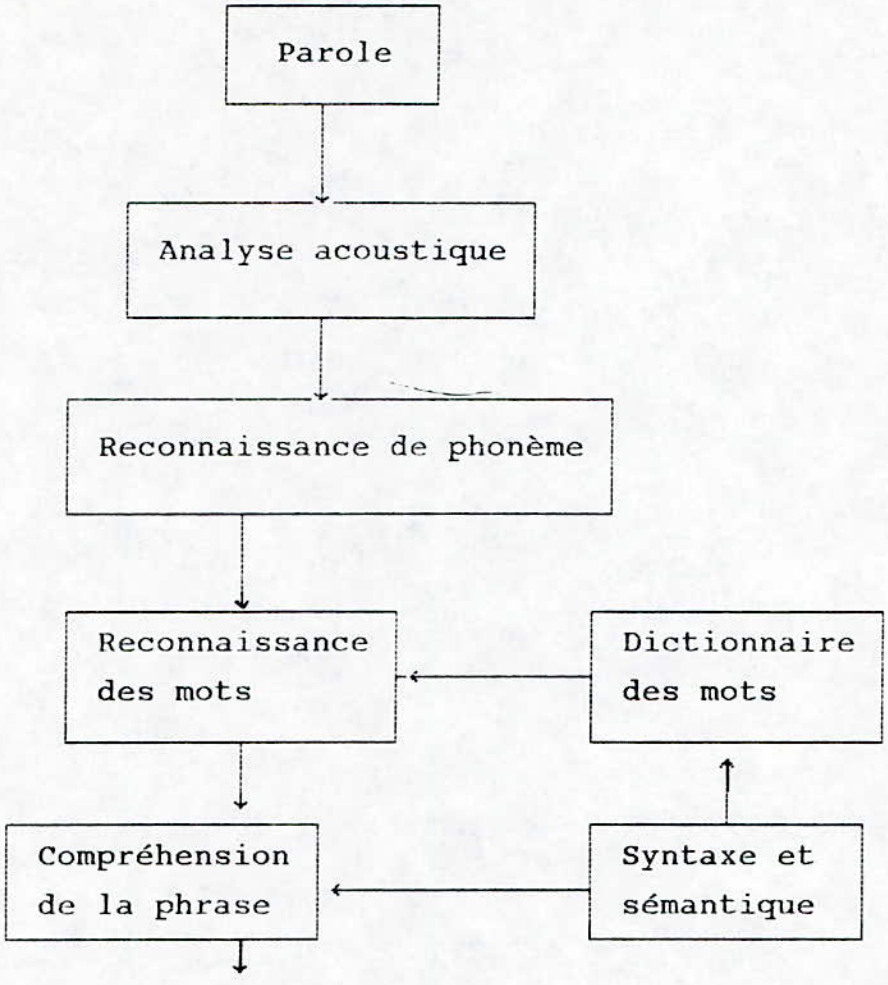


Fig-8 Système de reconnaissance de courtes phrases

3.3- PERSPECTIVES D'AVENIR ET POSSIBILITES ACTUELLES

Actuellement, pour améliorer le taux de reconnaissance, on applique les techniques de l'intelligence artificielle et des systèmes experts. De cette façon la reconnaissance s'effectue non seulement au niveau phonétique, mais aussi au niveau du lexique, de la syntaxe et du sens.

Depuis quelques années une nouvelle voie s'est ouverte avec l'apparition des réseaux neuro-mimétiques que des résultats spectaculaires ont fait connaître au grand public. Ces nouveaux systèmes vont conquérir à moyen terme une part respectable du marché du traitement de la parole et de l'image.

ANNEXES

A- NOTIONS DE LA THEORIE DES PROBABILITES

A-1. RAPPEL DES DEFINITIONS ET DES THEOREMES ELEMENTAIRES

a-Soient tous les événements mutuellement exclusifs d'un d'espace d'echantillonnage alors :

$$\sum_i P(A_i)=1$$

b-Probabilité conditionnelle : on considère deux événements A et B. La probabilité conditionnelle P(B/A) est la probabilité de réalisation de B sachant que A s'est réalisé.

c-probabilité composée:c'est la probabilité de réalisation simultanée des événements A et B :

$$P(A,B)=P(A).P(A/B)=P(B).P(B/A)$$

Pour plusieurs événements simultanées:

$$P(A_1,A_2,\ldots,A_n)=P(A_1).P(A_2/A_1).P(A_3/A_1,A_2)\ldots P(A_n/A_1..A_{n-1})$$

d-Evénements indépendants :deux événements sont dits indépendants si :

$$P(A/B)=P(A)$$

Théorème :Soient deux événements indépendants A et B alors :

$$P(A,B)=P(A).P(B)$$

Dans le cas de plusieurs variables indépendantes :

$$P(A_1,A_2,..A_n)=\prod_{i=1}^{n} P(A_i)$$

e-Théorème des probabilités totales:

On considère les événements H_i mutuellement exclusifs dont l'union est l'espace fondamental Ω. Si A est un événement quelconque alors:

$$P(A)=\sum_i P(H_i).P(A/H_i)$$

3-Théorème de Bayes

On prend les mêmes conditions que le théorème précédent:

$$P(H_i/A)=\frac{P(H_i).P(A/H_i)}{P(A)}$$

A-2 VARIABLE ALEATOIRE. LOI DE DISTRIBUTION:

1-Une variable X est dite aléatoire si elle prend ses valeurs selon une probabilité P(X).

Une variable aléatoire X peut être discrète ou continue.

Il est commode d'introduire la fonction de probabilité ou fonction de répartition $P(X=x)$.

Dans le cas d'une variable aléatoire discrète on a bien sûr:

$\sum_i P(X=x_i)=1$ (x_i:chacune des valeurs que peut prendre la variable aléatoire).

Pour une variable aléatoire continue on préfère définir une fonction densité de probabilité $p(x)$ telle que :

$$p(x)>0$$

$$\int_{-\infty}^{+\infty} p(x)\,dx=1$$

2-Loi normale ou loi de Gauss: $p(x)=e^{-(x-\mu)^2\sigma^2/2}/\sqrt{2.\pi}.\sigma$

μ: moyenne ou espérance de X; σ^2:variance

Loi normale multidimensionnelle pour des variables indépendantes:

$$p(x_1,\ldots x_n)=\prod_{i=1}^{n} e^{(x-\mu)_i/\sigma_i^2/2}/\sqrt{2.\pi}\,\sigma_i$$

B- ANALYSE DE QUELQUES SIGNAUX

On veut procéder à l'analyse de quelques signaux typiques, par exemple un pseudo bruit blanc , un signal sans bruit, un signal avec un bruit faible et un signal fortement bruité.

Conditions de l'analyse:

- Fréquence d'échantillonnage: f_e=10Khz
- Nombre d'échantillons de la fenêtre d'analyse pour la LPC: N=300
- Ordre de la prédiction linéaire: p=14
- Nombre de points pour la transformée de Fourier: N_f=128

Caractéristiques du signal :

Le signal que nous avons généré pour cet exemple est simplement la somme de trois sinusoïdes dont voici les fréquences :

f_1=275Hz ; f_2=1375Hz ; f_3=1650Hz

Pour les signaux faiblement et fortement, on prend le même signal auquel on ajoute respectivement une faible proportion du bruit et une forte.

L'analyse nous donne alors les graphes des figures 1,2,3 et 4. Sur les graphes ,en pointillés on représente le spectre de chaque signal et en trait la réponse impulsionnelle du filtre tous-pôles qui approxime ce signal.

Commentons ces graphes. Sur la figure 1, on montre le spectre et le modèle autorégressif d'un bruit blanc , on constate d'abord que les deux courbes ne se ressemblent pas et ensuite que la réponse du filtre présente à une fréquence élevée un pic du à la préaccentuation.
Sur la figure 2 les pics des deux courbes coïncident.
Sur la figure 3 on voit que le filtre approxime nettement plus mal le signal.
On observe sur la figure 4 une détérioration de la qualité de la prédiction.

Conclusion:
La modélisation autorégressive est très sensible au bruit.

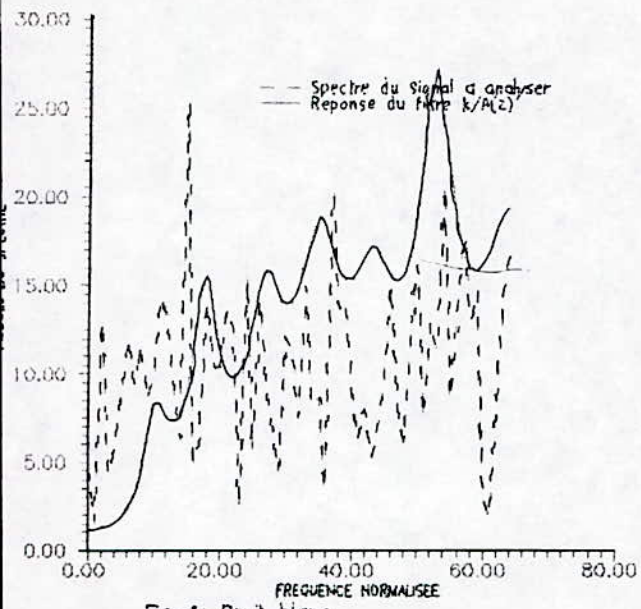


Fig-1: Bruit blanc

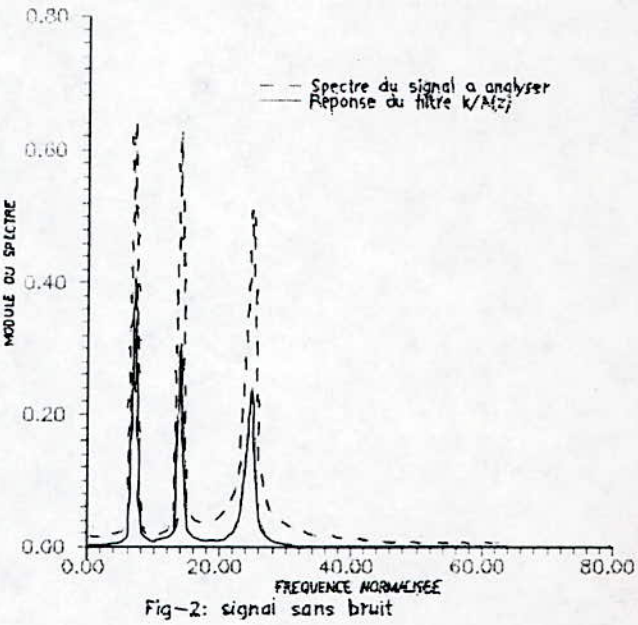


Fig-2: signal sans bruit

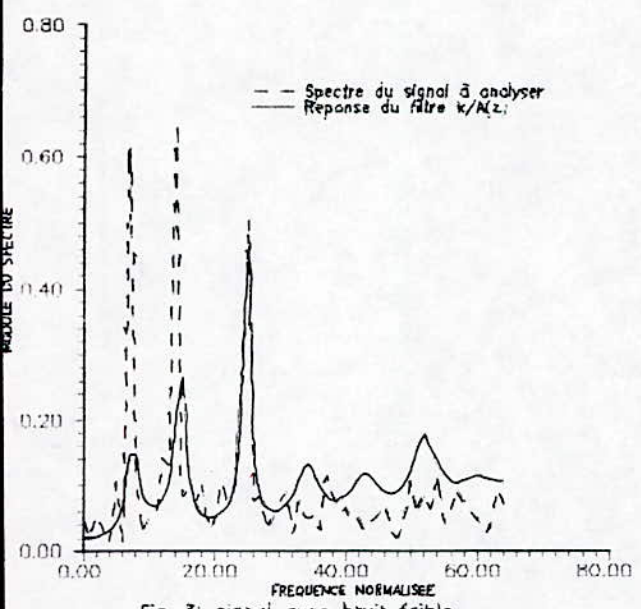


Fig-3: signal avec bruit faible

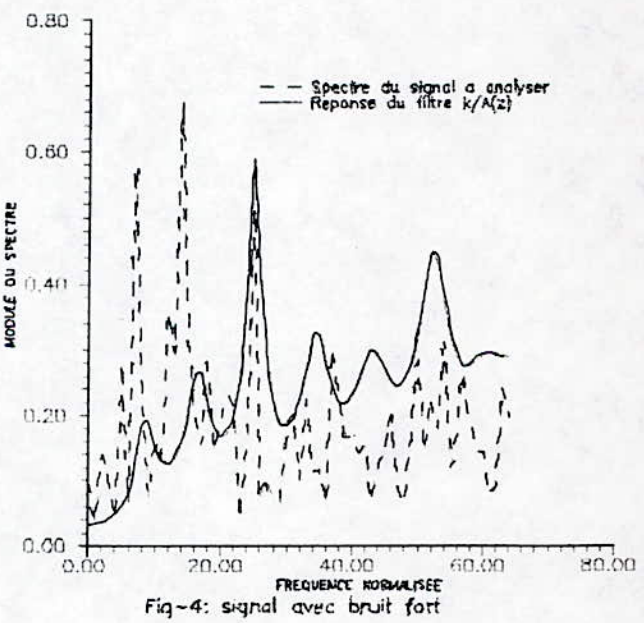


Fig-4: signal avec bruit fort

C-DESCRIPTION DES PROGRAMMES

Nous avons divisé nos en plusieurs groupes de sous-programmes, ou unite (dans la terminologie de Turbo-Pascal) .Voici les différentes unités:

- INFO :cette unité rassemble toute les déclarations communes à tous les programmes et procède aussi à l'initialisation de certaines variables.
- FOURIER : On y trouve les sous-programmes de calcul de la FFT et de la réponse du filtre.
- MODLPC :cette unité rassemble les programmes qui procèdent à l'analyse proprement dite du signal:préaccentuation, pondération, autocorrélation, résolution de la matrice de Toeplitz et même calcul du cepstre et de l'erreur de prédiction.
- FORMANT :elle procède à la recherche des formants selon deux méthodes :par résolution du polynôme prédicteur et par recherche des maximums du spectre.
- DISTANCE: grâce à cette unité on peut calculer quelques distances:euclidienne, cepstrale, de Itakura-saito ,de Itakura.

RECON: le programme de reconnaissance par la méthode des chaînes de Markov calcule simplement les probabilités Forward ou Backward.

APPREND: ici, on doit procéder à l'entrainement des modèles en présentant à chaque modèle différentes versions de chaque mot.

```
unit info;

interface

CONST
Pf2=128;  {nombre d'échantillon que traite la FFT}

N=200;    {nombre d'échantillon}

P=14;     {ordre de la prediction}

tetha=5;   {nombre d'états de la chaine de Markov}

Te=1E-4;Fe=1/Te; {periode et fréquence d'échantillonnage}

type
reel=extended;

vecteur=array [0..N] of reel;{données pour l'analyse LPC}

baton=array [0..P] of reel;  {coefficients de la LPC}

fleche=array [0..pf2-1] of reel;{données pour la FFT}

transition=array [0..tetha+1,0..tetha+1] of reel;
                              {matrice de transition de Markov}
emission=array [0..tetha+1] of  reel;{vecteur  d'émission  de
markov}

moments=array [0..tetha+1] of baton;

mot=^trim;
trim=record    {suite de vecteurs acoustiques qui donnent  un
mot}
          phon:baton;
```

```
            debut,fin,avant,apres:pointer;
      end;
stochas=^subir;
subir=record {suite des vecteurs d'émission  des  composantes
du mot}
        emis:emission;
        debut,fin,avant,apres:pointer;
         end;

modele=record
{cet enregistrement contient les  traitements  qu'on  a  fait
subir au signal
 vocal:autocorrelation,prediction,cepstre.Et  aussi  l'erreur
de prediction}
            a,rx,c:baton;
           sx2:reel;
           coreltest,lpctest,ceptest:boolean;
            end;

information=record
                 unitey,unitex,titre:string;
                 end;


var
zeroP:baton;
zeroN:vecteur;
zeroPF2:fleche;
zeroemis:emission;



implementation
procedure initialiser;
var i:integer;
```

```
begin
for i:=0 to p do zeroP[i]:=0;
for i:=0 to n do zeroN[i]:=0;
for i:=0 to pf2-1 do zeropf2[i]:=0;
for i:=0 to tetha+1 do zeroemis[i]:=0;
end;

   begin
   initialiser;
   end.
```

```
unit fourier;

interface
uses info;

procedure fft(xr1:fleche;var xr2: fleche);
{calcul la FFT}

procedure tfd(var xr1:baton;var xr2:fleche);
{calcul l'inverse de la rponse en frquence du filtre}

implementation

PROCEDURE FFT(xr1:fleche;var xr2:fleche);
var im,ni,i,j,m,v,kj,km,mt,l,sil,t,s:integer;
    xi1,xi2:fleche;
    xy:reel;
function inverse(L:INTEGER):INTEGER;
VAR M,I:INTEGER;
                        BEGIN
M:=0;
FOR I:=1 TO V DO BEGIN
M:=2*m+(L MOD 2);
L:=l div 2;       END;
INVERSE:=M;             END;

begin   {sous-programme FFT}
m:=pf2;xy:=2*pi/pf2;
v:=0;NI:=pf2;
FOR I:=0 TO pf2-1 DO XI1[I]:=0;
REPEAT
V:=V+1;
NI:=NI DIV 2;
UNTIL NI=1;
for i:=1 to v do begin
m:=m div 2;xr2:=xr1;xi2:=xi1;
for j:=0 to pf2-1 do begin
im:=(j xor m);kj:=0;km:=1;
if j>im then begin kj:=1;km:=0;end;
mt:=m;
t:=im div m;l:=0;
for s:=1 to i  do begin
l:=2*l+(t mod 2);
t:=t div 2;
end;
sil:=l;
XR1[im]:=XR2[j]*cos(xy*kj*mt*sil)+XI2[j]*sin(xy*kj*mt*sil)
+COS(xy*km*mt*sil)*XR2[im]+XI2[im]*sin(xy*km*mt*sil);
XI1[im]:=XI2[j]*cos(xy*kj*mt*sil)-XR2[j]*sin(xy*kj*mt*sil)
-XR2[im]*sin(xy*km*mt*sil)+XI2[im]*cos(xy*km*mt*sil);

end;end;
fOR I:=0 TO pf2-1 DO  begin
XR2[inverse(I)]:=sqrt(SQR(XR1[I])+SQR(XI1[I]));end;
END;
```

```
PROCEDURE TFD(var XR1:baton;VAR XR2:fleche);
var XI2:fleche;
   I,J:integer;
   x,y:reel;

   begin
   xr2:=zeropf2;
   XI2:=XR2;
   x:=2*pi/pf2;
   for i:=0 to Pf2-1 do begin
   for j:=0 to p do begin
   y:=i*j*x;
   XR2[i]:=xr2[i]+xr1[j]*cos(y);
   xi2[i]:=xi2[i]-xr1[j]*sin(y);
   end;
   end;
   for i:=0 to pf2-1 do xr2[i]:=sqrt(sqr(xr2[i])+sqr(xi2[i]));
   end;


  begin
  end.
```

```
unit modlpc;


interface

uses info;
PROCEDURE PREACCENTUATION(var x:vecteur);

function w(var i:integer):reel;

PROCEDURE PONDERATION(var x:vecteur);

PROCEDURE CORRELATION( x:vecteur;var rx:baton);

PROCEDURE TOEPLITZ( rx:baton;var a:baton);
{rsolution du syst
me d'quations}

procedure cepstre(var Ax:modele);

procedure errpredic(var ax:modele);

procedure lpc(var x:vecteur;var ax:modele);

 implementation

 PROCEDURE PREACCENTUATION(var x:vecteur);
VAR
I:INTEGER;
S:VECTEUR;
BEGIN
FOR I:=1 TO N DO begin
S[I]:=X[I]-0.95*X[I-1];
end;
x:=s;
END;

function w(var i:integer):reel;
var z:reel;
begin
z:=2*pi*I/N;
w:=0.54-0.46*cos(z);
end;

PROCEDURE PONDERATION(var x:vecteur);
var
i:integer;
begin
for i:=0 TO N do  begin
x[i]:=x[i]*w(i);
end;
end;
```

```
PROCEDURE CORRELATION( x:vecteur;var rx:baton);
var
rc:reel;
I,K,j:integer;
begin
for k:=0 to P DO BEGIN
rc:=0;j:=n-1-k;
for i:=0 to j  do  BEGIN rc:=x[i]*x[i+k]+rc;END;
rx[k]:=rc;
end;
end;


PROCEDURE TOEPLITZ(rx :baton;var a:baton);
var
m,I:integer;
M1,M2:BOOLEAN;
px,alpha,k:reel;
Am:array [BOOLEAN] OF BATON;

begin

M1:=FALSE;m2:=TRUE;
alpha:=rx[0];


FOR I:=0 TO P DO BEGIN
Am[FALSE,I]:=0;
Am[TRUE,I]:=0;
END;

Am[M1,0]:=1;Am[M2,0]:=1;

for m:=1 to p  do begin
   px:=0;
   m1:=NOT(M1);
   m2:=NOT(M2);
      for i:=0 to m-1 do BEGIN px:=px+Am[m2,i]*rx[m-i];END;
   k:=-px/alpha;

      FOR I:=1 TO M-1 DO BEGIN Am[M1,I]:=Am[M2,I]+K*Am[M2,M-I];END;

   Am[m1,m]:=k;
   alpha:=alpha*(1-k*k);
end;
A:=Am[M1];
end;
```

```
procedure cepstre(var Ax:modele);

var
i,k:integer;
x,y:reel;

begin
with ax do begin
FOR I:=0 TO P DO BEGIN C[I]:=0;END;
c[0]:=ln(rx[0]);
for i:=1 TO P DO BEGIN
X:=0;
 FOR K:=1 TO I-1 DO BEGIN
    X:=X+(1-K/I)*A[K]*C[I-K];
 END;
c[i]:=-A[I]-X;
end;
ceptest:=true;
end;
END;


procedure errpredic(var ax:modele);
var i:integer;
begin
with ax do begin sx2:=0;
for i:=1 to p do sx2:=-rx[i]*a[i]+sx2;
end;
end;


procedure lpc(var x:vecteur;var ax:modele);
begin
with ax do begin
preaccentuation(x);
ponderation(x);
correlation(x,rx);
toeplitz(rx,a);
coreltest:=true;
lpctest:=true;
end;
errpredic(ax);
End;

begin
end.
```

```
unit formant;


interface


uses info,fourier;


procedure FORMANTS(var af,fm,bp:baton);
{recherche les formants par calcul de la rponse frquentielle}
  {fm:liste des formants ;
         bp:liste des bandes passantes correspondantes}

procedure solution( a:baton;ct:baton);
{recherche les formants par rsolution du polynome de prdiction}
{ct:les indices impaires indiquent les formants,
    les indices  paires  indiquent les bandes passantes }


 implementation


procedure FORMANTS(var af,fm,bp:baton);
var i,j,k,m:integer;
x,y,x1,a,b,c,xi,delta,fp,ym,x0:reel;
  d:fleche;

begin
tfd(af,d);
j:=1;
fm:=zerop;bp:=zerop;
for i:=0 to (pf2 div 2) do d[i]:=1/D[i];
for i:=1 to ((pf2 div 2)-1) do begin
if ((d[i]>d[i-1]) and (d[i]>d[i+1])) then begin
c:=d[i];
a:=(d[i-1]+d[I+1])/2+c;
b:=(d[I+1]-d[I-1])/2;
x1:=-b/(2*a);
x:=i/2+x1;
c:=c-(a*x1*x1+b*x1+c)/sqrt(2);
delta:=sqr(b)-4*a*c;
fp:=abs(SQRT(abs(delta))/a);
fm[j]:=x*fe/pf2;
bp[j]:=fp*fe/pf2;
j:=j+1;if j>14 then j:=14;
end;
end;
end;
```

```
    procedure solution( a:baton;ct:baton);
    var  k,i,j:integer;
       c,d,e,si,ps,pt,s:reel;
       b,r:baton;
    begin
    k:=-1;
    repeat
    k:=k+1;
    until abs(a[p-k])>0.0001;
    k:=p-k;
    b:=a;
    for i:=0 to k do
    a[i]:=b[k-i];
    b:=zerop;

    j:=0;
    e:=0.01;
    repeat
 si:=-a[1]/a[0];
 ps:=a[2]/a[0];

 repeat
 s:=si;
 pt:=ps;
 b[0]:=a[0];
 b[1]:=a[1]+b[0]*s;
 for i:=2 to k do
 b[i]:=a[i]+s*b[i-1]-pt*b[i-2];
 r[0]:=0;r[1]:=b[0];
 for i:=2 to k do
 r[i]:=b[i-1]+s*r[i-1]-pt*r[i-2];
 d:=r[k]*r[k-2]-r[k-1]*r[k-1];
 si:=s-(b[k]*r[k-2]-b[k-1]*r[k-1])/d;
 ps:=pt-(b[k]*r[k-1]-b[k-1]*r[k])/d;
 until abs(si-s)+abs(ps-pt)<e ;

 if ((ps>=0) and (abs(si)<=2)) then begin
  d:=-si/2;if abs(d)<0.0001 then d:=pi/2 else d:=arctan(sqrt(1-d*d)/d);
 ct[j]:=sqrt(abs(ps));
 write(ct[j],' ');
 j:=j+1;
 ct[j]:=abs(d);
 writeln(ct[j]);
 j:=j+1;
 end;
 k:=k-2;
 for i:=0 to k do a[i]:=b[i];
 until k<2;
  end;



begin
end.
```

```
unit distance;

interface

uses info;


function euclide(var ax,bx:modele):reel;

 function DISTANCE_CEPSTRALE(var a,b:modele):reel;

 function itakura_saito(var a,b:modele):reel;

 function itakura(var_a,b:modele):reel;


 implementation

FUNCTION euclide(var ax,bx:modele):reel;
var i:integer;
   tx:reel;
begin
tx:=0;
for i:=0 to p do begin
tx:=tx+abs(ax.a[i]-bx.a[i]);
end;
euclide:=tx;
end;


 function DISTANCE_CEPSTRALE(var a,b:modele):reel;
 var i,j:integer;
  x:reel;
 begin
x:=0;
for i:=1 to p-1 do begin
x:=x+sqr(a.c[i]-b.c[i]);
end;
distance_cepstrale:=2*x+sqr(a.c[0]-b.c[0]);
end;

function itakura_saito(var a,b:modele):reel;
var i,j:integer;
    x1:reel;
    c1:baton;
begin
c1:=zerop;
for i:=1 to p do
for j:=1 to p do
c1[i]:=a.rx[abs(j-i)]*b.a[j]+c1[i];
x1:=0;
for i:=1 to p do
x1:=c1[i]*b.a[i]+x1;
itakura_saito:=x1/b.sx2-ln(a.sx2/b.sx2)-1;
end;
```

```
function itakura(var a,b:modele):reel;
var i,j:integer;
    x1:reel;
    c1:baton;
begin
c1:=zerop;
for i:=1 to p do
for j:=1 to p do
c1[i]:=a.rx[abs(j-i)]*b.a[j]+c1[i];

x1:=0;
for i:=1 to p do
x1:=c1[i]*b.a[i]+x1;
itakura:=(x1/a.sx2)-1;
end;

begin
end.
```

```
PROGRAM recon;

uses info;
Type grec=^etrusque;
   etrusque=record
   hun:transition;
   debut,fin,avant,apres:pointer;
   end;

var algorithme:char;

function forwards(var dune:stochas;var At:transition):reel;
var alpha,alpha1:emission;
i,j:integer;
x,Ptot:reel;
dil:stochas;

begin
dune:=dune^.debut;
for j:=1 to tetha do alpha1[j]:=At[0,j]*dune^.emis[j];
repeat
alpha:=alpha1;
dune:=dune^.apres;
for j:=1 to tetha do begin
x:=0;
for i:=1 to tetha do begin
x:=x+alpha[i]*At[i,j]*dune^.emis[j];
end;
alpha1[j]:=x;end;
dil:=dune^.apres;
until dil^.apres=nil;
Ptot:=0;
for i:=1 to tetha do Ptot:=Ptot+alpha1[i]*At[i,tetha+1];
forwards:=Ptot;
end;

nction backwards(var dune:stochas;var At:transition):reel;
```

```
        var beta,beta1:emission;
        i,j:integer;
        x,Ptot:reel;
        dil:stochas;

        begin
        dune:=dune^.debut;dune1:=dune^.fin;
        for j:=1 to tetha do beta1[j]:=At[0,j]*dune^.emis[j];
        repeat
        beta:=beta1;
        dune:=dune^.avant;
        for j:=1 to tetha do begin
        x:=0;
        for i:=1 to tetha do begin
        x:=x+beta[i]*At[i,j]*dune^.emis[j];
end;
        beta1[j]:=x;end;
        dil:=dune^.avant;
        until dil^.avant=nil;
        Ptot:=0;
        for i:=1 to tetha do Ptot:=Ptot+beta1[i]*At[i,tetha+1];
        backwards:=Ptot;
        end;

        {*****************************************************************
        *************}
        function automate(var mu,sivar:moments;var  At:transition;var
        verbe:mot):reel;
        var i,j:integer;
        alea:stochas;
        Ptot:reel;

        procedure gauss(var verbe:mot;var alea:stochas);
        var i,j,k:integer;
            x,y,z:reel;
  cop,flic,debutalea,finalea:pointer;
```

```
begin
y:=exp(-p/2*ln(2*pi));
new(alea);
alea^.avant:=nil;
debutalea:=alea;
z:=0;

repeat
for i:=1 to tetha do begin
x:=y; z:=0;
for j:=1 to p do begin
z:=z-0.5*sqr((verbe^.phon[j]-mu[i,j])*sivar[i,j]);
x:=x/sivar[i,j];
{x:=x*exp(-0.5*sqr((verbe^.phon[j]-mu[i,j])/sivar[i,j]))/sivar[i,j];}
end;
alea^.emis[i]:=x*exp(z);
end;

        alea^.debut:=debutalea;  cop:=alea;  new(alea);  flic:=alea;
      alea^.avant:=cop;  alea:=cop;  alea^.apres:=flic;  alea:=fli-
      c;  alea:=cop;
       verbe:=verbe^.apres;
       until verbe^.apres=nil;
       alea^.apres:=nil;
       finalea:=alea;
       alea:=debutalea;
       alea^.fin:=finalea;
       end;


      begin
      gauss(verbe,alea);
      case algorithme of
      'f','F':Ptot:=forwards(alea,At);
      'b','B':Ptot:=backwards(alea,At);
      end;

      end;


      {*****************************************************************
      ***************}

       begin
       end.
```

```
PROGRAM apprend;

uses info;
Type grec=^etrusque;
   etrusque=record
   hun:transition;
   debut,fin,avant,apres:pointer;
   end;

   robot=record
   mu,sivar:moments;
   At:transition;
   end;

var algorithme:char;
     bion:file of robot;
{*****************************************************************}

procedure     apprentissage(Var     mu,sivar:moments;var
at:transition;
Var mu1,sivar1:moments;Var at1:transition;Var verbe:mot);
var epsilon:grec;
gamma,alea,aline,betty:stochas;
  nulle:reel;
  debut,fin,avant,apres:pointer;
  i,j,l:integer;
  psi:transition;
  omega:emission;
  muo,sim,muo1,sim1:baton;

Procedure forwards(var dune,aline:stochas);
var alpha,alpha1:emission;
i,j:integer;
x,Ptot:reel;
dil:stochas;
```

```
avant,apres,debut,fin:pointer;

begin
new(aline);aline^.avant:=nil;debut:=aline;
dune:=dune^.debut;
for j:=1 to tetha do alpha1[j]:=At[0,j]*dune^.emis[j];
repeat
alpha:=alpha1;
dune:=dune^.apres;
for j:=1 to tetha do begin
x:=0;
for i:=1 to tetha do begin
x:=x+alpha[i]*At[i,j]*dune^.emis[j];
end;
alpha1[j]:=x;end;
dil:=dune^.apres;
aline^.emis:=alpha;
avant:=aline;
new(aline);
aline^.avant:=avant;
aline^.debut:=debut;
apres:=aline;
aline:=avant;
aline^.apres:=apres;
aline:=apres;
until dil^.apres=nil;
aline^.emis:=alpha1;
aline^.apres:=nil;
aline^.debut:=debut;
fin:=aline;
aline:=debut;
aline^.fin:=fin;
Ptot:=0;
for i:=1 to tetha do Ptot:=Ptot+alpha1[i]*At[i,tetha+1];
```

```
end;

procedure backwards(var dune,betty:stochas);
var beta,beta1:emission;
i,j:integer;
x,Ptot:reel;
dil:stochas;
avant,apres,debut,fin:pointer;
begin
new(betty);betty^.apres:=nil;fin:=betty;
dune:=dune^.debut;dune:=dune^.fin;
for j:=1 to tetha do beta1[j]:=At[0,j]*dune^.emis[j];
repeat
beta:=beta1;
dune:=dune^.avant;
for j:=1 to tetha do begin
x:=0;
for i:=1 to tetha do begin
x:=x+beta[i]*At[i,j]*dune^.emis[j];
end;
beta1[j]:=x;end;
betty^.emis:=beta;
apres:=betty;
new(betty);betty^.apres:=apres;betty^.fin:=betty;
avant:=betty;
betty:=apres;
betty^.avant:=avant;
betty:=avant;
dil:=dune^.avant;
until dil^.avant=nil;
betty^.emis:=beta1;
betty^.avant:=nil;betty^.fin:=fin;
debut:=betty;betty:=fin;betty^.debut:=debut;
```

```
Ptot:=0;
for i:=1 to tetha do Ptot:=Ptot+beta1[i]*At[i,tetha+1];
end;

procedure gauss(var verbe:mot;var alea:stochas);
var i,j,k:integer;
    x,y,z:reel;
f     cop,flic,debutalea,finalea:pointer;

 begin
 y:=exp(-p/2*ln(2*pi));
 new(alea);
 alea^.avant:=nil;
 debutalea:=alea;
 z:=0;

 repeat
 for i:=1 to tetha do begin
 x:=y; z:=0;
 for j:=1 to p do begin
 writeln(z,' ',j,' ',sivar[i,j],' ',verbe^.phon[j]);

 z:=z-0.5*sqr((verbe^.phon[j]-mu[i,j])/sivar[i,j]);

 x:=x/sivar[i,j];
 {x:=x*exp(-0.5*sqr((verbe^.phon[j]-
mu[i,j])/sivar[i,j]))/sivar[i,j];}
 end;
 alea^.emis[i]:=x*exp(z);
 end;

 alea^.debut:=debutalea;
 cop:=alea;
 new(alea);
 flic:=alea;
```

```
alea^.avant:=cop;
alea:=cop;
alea^.apres:=flic;
alea:=flic;
alea:=cop;


verbe:=verbe^.apres;
until verbe^.apres=nil;
alea^.apres:=nil;
finalea:=alea;
alea:=debutalea;
alea^.fin:=finalea;
end;

begin (*Programme Principal*)
gauss(verbe,alea);
forwards(alea,aline);
Backwards(alea,betty);

new(epsilon);debut:=epsilon;avant:=epsilon;
epsilon^.avant:=nil;aline:=aline^.debut;
betty:=betty^.fin;betty:=betty^.debut;
repeat
for i:=0 to tetha+1 do begin
for j:=0 to tetha+1 do begin
epsilon^.hun[i,j]:=at[i,j]*aline^.emis[i]*betty^.emis[j];
end;
end;
new(epsilon);epsilon^.debut:=debut;
epsilon^.avant:=avant;apres:=epsilon;
epsilon:=avant;epsilon^.apres:=apres;epsilon:=apres;
aline:=aline^.apres;
betty:=betty^.apres;
until betty^.apres=nil;
```

```
 fin:=epsilon;
 epsilon:=debut;
 epsilon^.fin:=fin;
new(gamma);debut:=gamma;avant:=gamma;
gamma^.avant:=nil;

repeat
for i:=0 to tetha+1 do begin
nulle:=0;
for j:=0 to tetha+1 do begin
gamma^.emis[i]:=epsilon^.hun[i,j]+nulle;
end;end;
new(gamma);
gamma^.debut:=debut;gamma^.avant:=avant;apres:=gamma;
gamma:=avant;gamma^.apres:=apres;gamma:=apres;
epsilon:=epsilon^.apres;
until epsilon^.apres=nil;
fin:=gamma;
gamma:=debut;
gamma^.fin:=fin;
epsilon:=epsilon^.debut;
 for i:=0 to tetha+1 do
 for j:=0 to tetha+1 do
 psi[i,j]:=0;

 repeat
 for i:=0 to tetha+1 do
 for j:=0 to tetha+1 do
 psi[i,j]:=psi[i,j]+epsilon^.hun[i,j];
 epsilon:=epsilon^.apres;
 until epsilon^.apres=nil;

 omega:=zeroemis;
 for i:=0 to tetha+1 do
 for j:=0 to tetha+1 do
```

```
 omega[i]:=omega[i]+psi[i,j];

 for i:=0 to tetha+1 do begin
 for j:=0 to tetha+1 do begin
  At1[i,j]:=psi[i,j]/omega[i];
 end;end;

 verbe:=verbe^.debut;
 gamma:=gamma^.debut;
 for       i:=0       to       tetha+1       do       begin
mu1[i]:=zerop;sivar1[i]:=zerop;end;
 repeat
 for i:=0 to tetha+1 do begin
 for l:=1 to p do begin
mu1[i,l]:=mu1[i,l]+verbe^.phon[l]*gamma^.emis[i]/omega[i];
sivar1[i,l]:=sivar1[i,l]+sqr(verbe^.phon[l]-mu[i,l])*gamma^.e
mis[i]/omega[i];
end;
end;
until verbe^.apres=nil;

end;
{*****************************************************************}


procedure entraine;
var i,j,k:integer;
    fire:file of baton;
    az:baton;
    verbe:mot;
    version:array[1..3] of mot;
    novoy,norob:string;
    debut,fin,apres,avant:pointer;
    mu,sivar:moments;
```

```
    At:transition;
    rob:robot;
    xs:reel;
  begin
  writeln('Programme d''entrainement des modeles');
  write('Nom du fichier voyelle:');
  readln(novoy);
  assign(fire,novoy);
  reset(fire);
  new(verbe);verbe^.avant:=nil;
  debut:=verbe;
  verbe^.debut:=debut;
  repeat
  read(fire,verbe^.phon);
  avant:=verbe;
  new(verbe);
  verbe:=debut;
  verbe^.avant:=avant;
  apres:=verbe;
  verbe:=avant;verbe^.apres:=apres;
  verbe:=apres;
  until eof(fire);  close(fire);
  verbe^.apres:=nil;fin:=verbe;verbe:=debut;verbe^.fin:=fin;

  for j:=0 to tetha+1 do begin
   for k:=0 to p do   begin
  mu[j,k]:=0;sivar[j,k]:=1; end;                    end;

  for j:=1 to tetha do
  for k:=1 to tetha do
  At[j,k]:=1/tetha;
 writeln('Attendre');
  At[0,0]:=0; at[0,tetha+1]:=0;
  for j:=1 to tetha do  At[0,j]:=1/tetha;
```

```
        for j:=1 to tetha-1 do At[j,tetha+1]:=0;
        At[tetha,tetha+1]:=1;At[tetha+1,tetha+1]:=1;
repeat
        xs:=At[1,1];


    apprentissage(mu,sivar,At,mu,sivar,At,verbe);

        funtil abs(xs-At[1,1])<0.001;
        rob.mu:=mu;
        rob.sivar:=sivar;
        rob.At:=At;
        writeln(novoy);
        writeln('vous sauvez le modele stochastique sur le
        fichier(.sto):');
        readln(norob);
        assign(bion,norob);
        write(bion,rob);
        close(bion);

        end;


        begin
        entraine;
         end.
```