Ecole Polytechnique, Promotion X 97

Rapport de Stage d'Option scientifique

Vincent ARSIGNY

Modélisation par champ de Markov du signal de parole et application à la reconnaissance vocale.

Option Image et Signal
Département de Mathématiques appliquées
Directeurs de l'Option : MM Stéphane MALLAT et Emmanuel BACRY

Stage effectué à l'Ecole Nationale Supérieure des Télécommunications de Paris, sous la direction de MM Marc SIGELLE et Guillaume GRAVIER, du 1^{er} *avril* 2000 *au* 7 *juillet* 2000

Table des matières

Abstract

This study focuses on an original modelling of speech, based on Random Markov Fields, originally used in Image Processing. This modelling was conceived so as to be applied to the automatic recognition of isolated words, which was actually performed during the stay at the ENST with a computer program written in C. One concentrated on this occasion on studying the relevancy of this type of modelling in speech recognition, and on optimizing it as much as possible.
In this repport may be found both a general presentation and a brief theoretical analysis of the most important mathematical tools used : in the first part, one gives the fundamentals of speech recognition via the *maximum of likelihood* approach, in the second one that of *Hidden Markov Models*, then that of Random Markov Fields, and in the two following parts, that of both phases of parameters estimation and recognition. Several personnal contributions of the author are detailled : an analysis of the properties of the *constrained states space*, a general strategy for calculus in the *uniform* constrained states space, and finally (in part 5), an exact formula for one type of partition function.
The sixth part yields a synthesis of the numerous results gathered through extensive numerical experiments. Thus, we see that our model performs speech recogntion in an acceptable way, and even a promissing one in the case of noise-corrupted signals. A good and quite general initialization for the model's parameters is given, along with other experimental rules for optimizing our model. And finally, a succinct study of the convergence of the EM algorithm puts into light the acute difficulty of the estimation of a certain sort of parameters in our model.

Résumé

L'objet la présente étude est une modélisation originale de la parole, utilisant les champs Markovien, qui sont un outil issu du traitement de l'image. Cette modélisation particulière a été développée pour être appliquée à la reconnaissance de mots isolés, ce qui fut fait au cours du stage d'option dans un logiciel programmé en C. L'objectif du stage consistait à s'assurer de la viabilité de l'utilisation de ce modèle en reconnaissance vocale, et de rendre cette dernière aussi efficace que possible.
Dans ce rapport figurent une présentation et une étude théorique des principaux outils utilisés : dans la première partie, on s'intéresse à la reconnaissance vocale par maximum de vraissemblance ; dans la deuxième à l'outil peut-être le plus répandu en reconnaissance : les *chaînes de Markov cachées*, dans la suivante aux champs de Markov, et dans les deux suivantes sont présentés les déroulements respectifs des phases d'estimation des paramètres (algorithme de type EM) et de reconnaissance (calcul détaillé du *score acoustique*). Plusieurs contributions originales de l'auteur de ce rapport sont exposées : une analyse des propriétés fondamentales de l'*espace contraint*, une méthode systématique de calcul de type combinatoire dans l'espace contraint *uniforme*, et plus loin (partie 5), un calcul exact de fonction de partition.
La $6^{\text{ième}}$ partie est consacrée à une synthèse des nombreux résultats obtenus au fil des nombreuses expérimentations numériques. On voit ainsi que les performances du modèle sont correctes pour des signaux de bonne qualité, et même encourageantes pour des signaux bruités. Une initialisation performante et simple des paramètres du modèle est également donnée dans un cas particulier, mais assez général ; d'autres résultats sur l'optimisation du modèle viennent compléter cette initialisation. Enfin, une étude expérimentale succinte de la convergence de l'algorithme EM est

présentée, mettant en valeur la grande difficulté de l'estimation d'un certain type de paramètres du modèle étudié.

Introduction

L'étude de la parole est un sujet très délicat. Au cœur même de la vie humaine, ce mode de communication n'a cessé de révèler de nouvelles subtilités, à mesure qu'avec le progrès des sciences et des techniques de nouvelles approches n'ont cessé de se multiplier. Avec la parole, on touche à un phénomène aléatoire exemplaire : reconnaissable et intelligible immédiatement pour une oreille humaine exercée, ce type de signal sonore est pourtant diablement difficile à modéliser mathématiquement. Il ne cesse de s'infléchir, de se déformer au gré des circonstances, des jeux de l'articulation, et des conditions climatiques...

De nombreux types de modélisations sont possibles, chacune ayant son domaine d'application privilégié, ses qualités, ses défauts rédibitoires dès que l'on sort de son domaine de pertinence. Durant le stage d'option dont les résultats sont mis en forme dans le présent rapport, on s'est intéressé à un modèle de parole original, appelé à servir d'outil pour la mise au point d'un logiciel de reconnaissance vocale spécialisé dans la reconnaissance de mots isolés. Ce modèle tire son originalité dans son histoire : il met à profit un outil mathématique mis au point dans la deuxième moitié du vingtième siècle pour le traitement de l'image, à savoir les champs de Markov.

L'objectif du stage, tel qu'il avait été défini par Marc Sigelle, Maître de Conférence à l'ENST, était double : il s'agissait d'étudier les propriétés théoriques du modèle afin de pourvoir le mettre en œuvre lors de la programmation d'un logiciel dédié à la reconnaissance vocale, devant être aussi efficace que possible. Il serait alors possible de commencer à formuler une réponse concernant tout d'abord la faisabilité, et ensuite la pertinence de cette nouvelle modélisation de la parole pour la reconnaissance vocale.

Nous verrons dans cette étude de quelle manière le problème a été abordé, aussi bien sur le plan théorique que pratique.

1 Le problème général de la reconnaissance de la parole

Dans cette partie, nous présentons les principes clés de la reconnaissance vocale de mots isolés. Notamment, on y aborde la notion fondamentale de processus aléatoire caché. Le point de vue adopté ici est celui du *maximum de vraissemblance.*

1.1 La paramétrisation du signal de parole

La parole est un type de signal présentant une grande variabilité, aussi bien temporellement que fréquentiellement. Et ceci pour toutes sortes de raisons : un individu parlera lentement ou rapidement selon les circonstances, articulera avec précision ou non, se situera dans un environnement parfaitement calme et exempt de tout bruit parasite, ou au contraire se trouvera dans un hall de gare particulièrement bruyant qui rendra confus les mots prononcés. C'est pour cela qu'un cadre probabiliste est bien adapté à la description du signal de parole.

Avant de préciser le cadre probabiliste de notre étude de la parole, commençons par une notion importante : la paramétrisation du signal considéré. En effet, pour les fins de la reconnaissance vocale, une question importante se pose : sur quel type de signal travailler ? Il est envisageable de travailler directement sur le signal discret $\{A_t\}_{t\in 1\dots T}$, obtenu par échantillonage (par exemple à 8 kHz, ce qui convient pour la voix humaine) d'un signal continu $A(t)$. Mais cette représentation possède un défaut majeur : en présence d'un bruit coloré, c'est-à-dire sélectif en fréquences, tous les coefficients sont affectés. Pour cette raison, on préfère généralement effectuer une analyse temps/fréquence du signal afin de représenter indépendemment les unes des autres un certain nombre de bandes de fréqences. On appelle *paramétrisation* cette nouvelle représentation parce qu'elle rend compte du signal autrement que par la donnée de ses échantillons.

La paramétrisation par banc de filtres Cette paramétrisation est un type particulier de représentation temps/fréquence du signal, bien adaptée à l'objectif de la modélisation de la parole pour la reconnaissance vocale. Nous présentons ici les différentes étapes de cette représentation, utilisée pour les les fins de cette étude.

1. **découpage du signal en fenêtres** par multiplication par une fonction en cloche à support compact, suffisamment régulière. Les différentes cloches utilisées ont un support de diamètre à peu près égal à 20ms et sont obtenues par une translation multiple de 10ms d'une cloche "mère". Une fenêtre correspond ainsi à une durée d'environ 10ms
2. **filtrage de chaque fenêtre** par un filtre permettant de réduire l'influence de l'excitation du conduit vocal par la glotte. En effet, sur un intervalle de temps relativement court, on peut décrire le signal de parole par : $S(t) = E \star CV(t)$, où $E(t)$ est le signal produit par la glotte, et où $CV(t)$ est la réponse impulsionnelle du conduit vocal. Or, le paramètre caractéristique du son produit est la forme prise par le conduit vocal, et non l'excitation par la glotte. Afin de réduire l'influence dans le signal de l'excitation glottale, riche en basses fréquences et pauvre en hautes fréquences, on réhausse les hautes fréquences de la fenêtre en la filtrant en conséquence.
3. **pasage dans le domaine fréquentiel** pour ce faire, une transformation de fourier discrète est effectuée sur le signal.

4. **analyse en fréquences** le spectre est divisé en 24 bandes de même largeur. On obtient une série de 24 coefficients en intégrant le module de la transformée de fourier du signal multiplié par des fonctions à support fini, que l'on déduit les unes des autres par translation. Ces coefficents rendent comptent de l'"énergie" (dans le sens de la norme $\mathcal{L}^1$) du signal dans les 24 bandes de spectre.
5. **passage au logarithme** les coefficients de la paramétrisation se déduisent des coefficients de l'étape précédente par passage au logarithme.

Au final, pour un mot de longueur 500ms, sa paramétrisation compte $\frac{500}{10} \times 24 = 1200$ paramètres. Une réalisation d'un signal de parole pour un mot donné se présentera ainsi comme une suite $(Y_t^k)_{t\in 1...T,\ k\in 1...24}$ avec $T < 100$ en général.

1.2 Une modélisation par processus cachés

Nous allons donner ici le cadre général de la modélisation du signal de parole utilisée pour la reconnaissance vocale. Classiquement, on considère que la réalisation (Y_t^k) n'est obtenue qu'indirectement, par l'intermédiaire de ce que l'on appelle un "processus caché", dépendant lui aussi du temps, que nous noterons (X_t^k). L'idée principale est que l'on ne connaît pas directemnt $P((Y_t^k) = (y_t^k))$ mais que l'on connait bien la loi conditionnelle $P((Y_t^k) = (y_t^k)|(X_t^k) = (x_t^k))$ et que la loi de (X_t^k) est connue. Le schéma suivant résume cette modélisation :

FIG. 1 – Modélisation du signal de parole par processus caché

Dans ce cadre, la probabilité de (Y_t^k) est obtenue grâce à la formule des probabilités totales :

$$P((Y_t^k) = (y_t^k)) = \sum_{(x_t^k)} P((Y_t^k) = (y_t^k)|(X_t^k) = (x_t^k))P((X_t^k) = (x_t^k))$$

Ainsi, la loi de probabilité de l'observation (Y_t^k) dépend des "états cachés" que le système adopte dans chaque bande au cours du temps. Dans le cadre de cette étude, une dépendance simple à été retenue : on a fait l'hypothèse que Y_t^k ne dépend que de X_t^k, et l'on suppose les $(Y_t^k \mid (X_t^k)\ )$ indépendants. Ainsi, pour un état caché global donné, les observations sont indépendantes les unes des autes dans ce cadre. Et la probabilité précédente devient :

$$\boxed{P((Y_t^k) = (y_t^k)) = \sum_{(x_t^k)} (\prod_{t,k} P(Y_t^k = y_t^k|X_t^k = x_t^k))\ P((X_t^k) = (x_t^k))}$$

Au paragraphe suivant, nous préciserons le type de processus caché retenu. Mais notons dès à présent que la durée du signal lui-même constitue également un processus aléatoire : comme on ne connaît pas avec certitude la durée du signal correspondant par exemple au mot "annulation",

on peut la modéliser par une variable aléatoire T qui est tirée avant le processu caché principal (X_t^k) : la modélisation de notre signal fait donc intervenir plus de deux niveaux de processus. Mais le processus de tirage au sort de la durée du signal n'est pas à proprement parler un processus "caché" : la durée du signal est parfaitement connue ! Ce qui ne sera pas le cas pour les processus cachés *stricto sensu*, qui devront être reconstitués par des méthodes probabilistes.

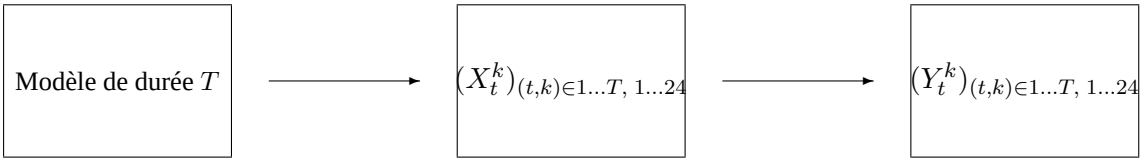

FIG. 2 – Incorporation d'une durée aléatoire dans la production de parole

Par soucis de simplicité et de clareté, nous omettrons cet élément la plupart du temps, car il n'a partiquement pas une grande importance, comme nous le montrerons par la suite. Mais n'anticipons pas.

1.3 Le principe du maximum de vraissemblance

Muni de notre cadre probabiliste à deux niveaux de variables aléatoires,(X_t^k) et (Y_t^k), nous pouvons énoncer le principe théorique fondamental de la reconnaissance vocale : le "maximum de vraissemblance" (en anglais, "maximum of likelihood" ou encore ML). Pour ceci, définissons un *dictionnaire* : $\{W\}$, l'ensemble des mots pouvant être reconnus. Pour chacun des mots w, l'on suppose que l'on connait $P(X_t^k = x_t^k|W = w)$ et $P(Y_t^k = y_t^k|X_t^k = x_t^k, W = w)$. L'on dira que pour une réalisation du signal de parole donné (y_t^k), le mot (ou les mots) w "reconnu"(s) par maximum vraissemblance est (sont) donné (s) par :

$$w = arg\ \max_w\ P(W = w|(Y_t^k) = (y_t^k))$$

Il s'agit donc du mot "le plus probable" conditionnellement à l'observation (Y_t^k) qui vient d'être faite. On peut encore réécrire cette formule de la manière suivante :

$$w = arg\ \max_w\ P((Y_t^k) = (y_t^k)|W = w)\frac{P(W = w)}{P((Y_t^k) = (y_t^k)} \quad (1)$$

$$= arg\ \max_w\ \underbrace{P((Y_t^k) = (y_t^k)|W = w)}_{score\ acoustique}\ \underbrace{P(W = w)}_{score\ linguistique} \quad (2)$$

Deux termes de natures distinctes apparaissent ainsi dans la quantité à maximiser : d'une part $P(Y_t^k) = (y_t^k)|W = w)$, nommé *score acoustique* , puisqu'il s'agit de la vraissemblance de l'observation (Y_t^k) conditionnellement au mot w. Et d'autre part le *score linguistique* $P(W = w)$. L'objet principal de notre étude sera justement l'estimation de ce premier terme, et non pas du deuxième puisqu'il ferait intervenir un modèle linguistique de la parole, stiplulant les fréquences d'apparition des mots du dictionnaire. Comme le contexte d'utilisation du dictionnaire reste ici indéfini, il n'y aurait pas grand sens à définir une loi de probabilité sur W non-uniforme.

On notera que cette modélisation ne satisferait pas aux besoins de la reconnaissance vocale à flot continu de parole : le signal considéré serait une succession de mots, et non la réalisation d'un unique mot. Il n'est question ici qu'uniquement de *mots isolés.*

1.4 Expression du score acoustique

Précisons à présent l'expression du score acoustique d'après nos hypothèses. En le notant LS et en tenant compte de l'indépendance mutuelle des $\{Y_t\}$, il vient :

$$LS = \prod_{t,k} P((Y_t^k) = (y_t^k) | W = w)$$

ce qui donne, en faisant intervenir le processus caché (X_t^k) :

$$\boxed{LS = \sum_{(x_t^k)} (\prod_{t,k} P(Y_t^k = y_t^k | X_t^k = x_t^k, W = w))\ P((X_t^k) = (x_t^k) | W = w)}$$

La forme prise par cette expression a une importance capitale : elle sera utilisée à de très nombreuses reprises dans ce qui suit. Dans ce cadre, modéliser la parole reviendra à préciser la forme prise par $P(Y_t^k = y_t^k | X_t^k = x_t^k, W = w)$ qui est la probabilité d'obtenir l'observation y_t^k à l'instant t dans la $k^{\text{ième}}$ bande pour une segmentation (états cachés) donnée, ainsi que l'expression de $P(X_t^k = x_t^k | W = w)$, probabilité *a priori* d'être dans l'état x_t^k à l'instant t et dans la bande k.

Approximation de Viterbi Pratiquement parlant, il n'est pas possible de calculer la somme de cette série. En effet, on ne dispose pas d'expression plus compacte de cette somme, et la sommation sur l'ensemble des (x_t^k) admissibles ne peut se faire même avec l'aide d'un calculateur, en raison du cardinal extrêment important de l'ensemble des configurations possibles. C'est pourquoi l'on est contraint de recourir à une approximation dans la pratique de la reconnaissance vocale.
L'approximation de Viterbi consiste à ne prendre en compte qu'un seul des termes de cette somme, à savoir celui qui correspond au (x_t^k) le plus probable pour la loi a posteriori (c'est-à-dire connaissant le signal (Y_t^k)). On fait l'hypothèse que celui-ci est unique. Nous le noterons $\underset{w}{x^*}$. Autrement dit, l'approximation se présente comme suit :

$$\begin{cases} \underset{w}{x^*} = arg \max\limits_{(x_t^k)} P((X_t^k) = (x_t^k) \mid (Y_t^k) = (y_t^k),\ W = w) \\ LS \simeq \underbrace{(\prod\limits_{t,k} P(Y_t^k = y_t^k \mid X_t^k = \underset{w}{x^*}{}_t^k,\ W = w))}_{\text{score d'attache aux données}} \underbrace{P((X_t^k) = \underset{w}{x^*} | W = w)}_{\text{score de segmentation}} \end{cases}$$

On voit ainsi apparaître deux le produit de 2 scores d'origines différentes : le score d'attache aux données, qui mesure la vraissemblance de notre réalisation pour une segmentation donnée (x^*), et un deuxième score, représentant la vraissemblance a priori de la segmentation x^* pour le mot w.

2 Un premier modèle de processus caché : les chaînes de Markov

Dans cette section, nous présentons les propriétés principales des chaîndes de Markov utiles pour notre étude. Au deuxième paragraphe, on donne une présentation originale d'un espace de configurations possible, *l'espace contraint*. Enfin, le dernier paragraphe expose des résultats originaux obtenus durant le stage, permettant le calcul explicite de nombreuses grandeurs statistiques dans un cas particulier d'espace contraint. Pour plus de résultats sur les HMM, on pourra consulter [2].

2.1 Les HMM, un outil classique

Utilisation des chaînes de Markov stationnaires Pour les besoins de la reconnaissances de la parole, le modèle le plus répendu est indéniablement celui des chaînes de Markov cachées, appelées communément HMM (pour Hidden Markov Model). L'ensemble du travail effectué dans ce chapitre se fait sur une bande de fréquence, dont on omettra de préciser le numéro. Notre premier modèle fait ainsi l'hypothèse de l'indépendence mutuelle des bandes de fréquences, ce qui sera corrigé au chapitre 3.

La propriété de Markov faible Dans le cadre de cette modélisation, à tout instant t, le système producteur du signal de parole se trouve dans un état x_t appartenant à un espace d'états que nous noterons E. En pratique, E est fini, et on l'assimilera à $1 \dots N$. Comme l'indique le nom du modèle, la dynamique du passage d'un état à un autre se fait de manière markovienne, plus précisément au sens des chaînes de Markov stationnaires :

$$P(X_{t+1} = x_{t+1} | X_1 = x_1, \dots, X_{t-1} = x_{t-1}, X_t = x_t) = P(X_{t+1} = x_{t+1} | X_t = x_t) \tag{3}$$

$$= p_{x_t, x_{t+1}} \, , \forall (x_1, \dots, x_t, x_{t+1}) \in E^{t+1} \, , \forall t \in 1...T \tag{4}$$

Cette propriété (dite de "Markov faible") signifie simplement que l'évolution du système de l'instant t à l'instant suivant, $t+1$, en supposant que l'on ait une parfaite connaissance de son évolution antérieure, ne dépend que de l'état présent x_t : peut importe le passé. L'hypothèse de stationnarité stipule ces transitions se font indépendamment de l'instant courant t. Aini, nous noterons $p_{i,j}$ la probabilité pour le système de passaer à l'état j à l'instant $t+1$ lorsqu'il est dans l'état i à l'instant courant.

Propriétés principales

1. **Expression des probabilités conjointes** : en appliquant la formule de Bayes un nombre suffisant de fois, on obtient :

$$P(X_t = x_t, X_{t-1} = x_{t-1}, \dots, X_1 = X_1) = P(X_t = x_t, X_{t-1} = x_{t-1}, \dots, X_2 = x_2 | X_1 = x_1) P(X_1 = x_1)$$

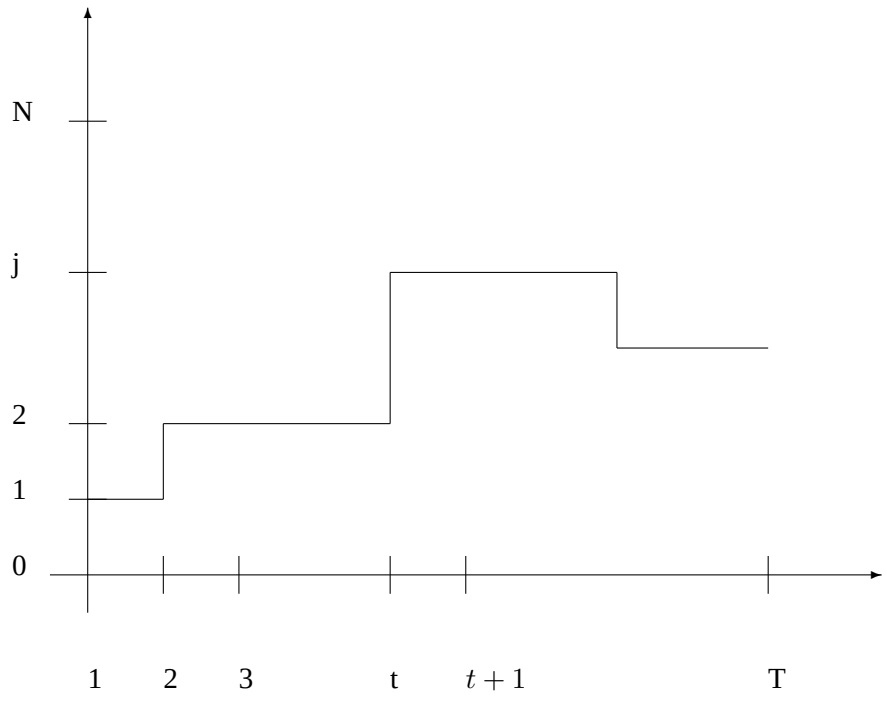

FIG. 3 – Exemple d'évolution Markovienne entre 1 et T

$$= P(X_t = x_t, X_{t-1} = x_{t-1}, \ldots, X_3 = x_3 | X_2 = x_2) P(X_2 = x_2 | X_1 = x_1) P(X_1 = x_1) \quad (5)$$

$$= \ldots \quad (6)$$

$$= P(X_1 = x_1) \prod_{a=2\ldots t} P(X_a = x_a | X_{a-1} = x_{a-1}) \quad (7)$$

$$= P(X_1 = x_1) \prod_{a=2\ldots t} p_{x_{a-1}, x_a} \quad (8)$$

D'où la formule très utilisée :

$$\boxed{P((X_a)_{a=1\ldots t} = (x_a)_{a=1\ldots t}) = P(X_1 = x_1) \prod_{a=2\ldots t} p_{x_{a-1}, x_a}}$$

2. Il est également utile de définir la **matrice de transition** $\mathbb{P}$ de la manière suivante :

$$\mathbb{P} = \begin{pmatrix} p_{1,1} & \cdots & p_{1,N} \\ \vdots & & \vdots \\ p_{N,1} & \cdots & p_{N,N} \end{pmatrix}, \text{ où } N = \text{Card}(E)$$

Si l'on note $\mathcal{V}_1$ la distribution de probabilités sur X_1, c'est-à-dire :

$$\mathcal{V}_1 = (P(X_1 = 1), \ldots, P(X_1 = N))$$

alors, la distribution de probabilités de X_t, $\mathcal{V}_t$, s'écrit :

$$\boxed{\mathcal{V}_t = \mathcal{V}_1 \times \mathbb{P}^{t-1}\ , \forall t \in 1 \dots T}$$

Nous aurons l'occasion d'utiliser cette propriété fondamentale par la suite.

Phénomène physique modélisé par les HMM La parole est un signal sonore très particulier : comme outil de communication, il est porteur de sens et est soumis à des règles assez strictes. Elle a ainsi une structure caractéristique : c'est une sucession de sons, et ces sons ont des caractéristiques quasiment stationnaires, sur des durées de l'ordre de 50 ms. D'un son à l'autre, ces caractérisques changent très nettement : on différencie ainsi facilement un son voisé (voyelles) comme le [a] d'un son non-voisé (consonnes) comme le son [t]. Le processus markovien caché que l'on utilise en reconnaissance de la parole modélise précisemmment cette succession de sons. Son caractère aléatoire permet de rendre compte de deux éléments caractéristiques du signal de parole :

* la variabilité des durées de chacun des sons coins.
* l'existence de sons longs ou courts

Autrement dit, le modèle markovien fournit un *modèle de durée* pour chacun des sons. Plus précisément, soit $i \in E$. Supposons que $X_1 = i$, et notons T_i la durée du séjour dans l'état i. En utilisant la propriété de Markov, on a immédiatement :

$$P(T_i = 1) = P(X_2 \neq i | X_1 = i) = 1 - p_{i,i} \tag{9}$$
$$P(T_i = 2) = p_{i,i}\ (1 - p_{i,i}) \tag{10}$$
$$\dots = \dots \tag{11}$$
$$P(T_i = L) = p_{i,i}^{L-1}\ (1 - p_{i,i}) \tag{12}$$

Ainsi T_i suit une loi géométrique de raison $p_{i,i}$, et son espérance $\mathrm{E}(Ti)$ est égale à $\frac{1}{1-p_{i,i}}$. A un état long correspondra ainsi un $p_{i,i}$ grand, et inversement. Il est donc possible de rendre compte de la durée des différents sons composant un mot en jouant sur les coefficients de $\mathbb{P}$.

Un exemple réel Pratiquement, on utilise deux états pour décrire chaque son, le premier pour son "début", et l'autre pour sa "fin". Pour le mot "guide", on utilise donc une représentation à 6 états. Dans le cadre d'un modèle "gauche-droite" et multibande, voici la matrice de transition obtenue pour la 8 ème bande de fréquences :

$$\mathbb{P} = \begin{pmatrix} 0.67 & 0.33 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.67 & 0.33 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.68 & 0.32 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.29 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.71 & 0.29 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.71 & 0.29 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

D'un simple coup d'oeil, on voit que les trois derniers états sont plus longs que les trois premiers, ce qui signifie que le troisième son de "guide", le [d] dure plus longtemps que ses prédécesseurs. On remarquera ici qu'un 7ème état "fictif", absorbant, a été ajouté, et nous reviendrons sur la nécessité de le faire intervenir au paragraphe suivant.

2.2 Modèle gauche-droite dans un espace contraint

Le modèle "gauche-droite" Dans cette étude, nous nous sommes intéressés à un cas particulier de chaîne de Markov, à savoir le cas où l'on n'autorise uniquement les transition de i vers i ou bien vers $i+1$. Ainsi, la suite (X_t), suite aléatoire, est croissante presque sûrement, au sens où $P((X_t) \text{ non croissante}) = 0$. La matrice de transition $\mathbb{P}$ prend alors une forme particulièrement simple :

$$\mathbb{P} = \begin{pmatrix} p_1 & 1-p_1 & 0 & 0 & \dots & 0 \\ 0 & p_2 & 1-p_2 & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & & \vdots \\ \vdots & & & & & \\ \vdots & & & \ddots & p_N & 1-p_N \\ 0 & \dots & & & 0 & 1 \end{pmatrix}$$

Cette matrice est ainsi bidiagonale supérieure : en-dehors de se diagonale et de sa surdiagoanle, tous ses coefficients sont nuls.

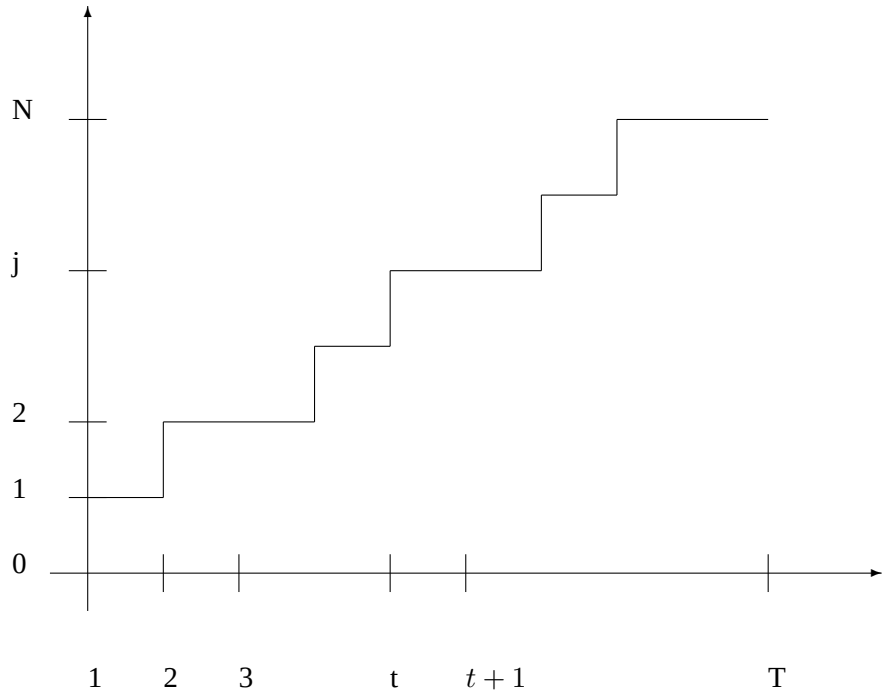

FIG. 4 – Exemple d'évolution "gauche-droite" entre 1 et T

Notons qu'ici encore, nous avons introduit un $N+1^{\text{ième}}$ état supplémentaire, que l'on suppose aborbant. Pourquoi doit-on l'utiliser ? Commençons par remarquer que si l'on veut se restreindre à une chaîne de Markov gauche-droite, le dernier terme de la diagonale est nécessairement égal à 1 puisque la matrice est par définition stochastique :

$$\sum_i \mathbb{P}(N+1,i) = 1 = \mathbb{P}(N+1,N+1)$$

Il n'est donc pas possible de se passer de ce "1" final. Cependant, puisque notre modèle doit comporter N états exactement par hypothèse, quid de ce $N+1^{\text{ième}}$ état ? Nous allons voir que la réponse tient à une autre hypothèse : celle d'un temps T fixé *a priori* pour le modèle. Par opposition à l'utilisation classique des chaînes de Markov sur un temps arbitrairement grand, nous travaillons ici sur un intervalle fixe et non sur un intervalle aléatoire borné supérieurement par un temps d'arrêt. Et pour résoudre la difficulté posée par cette approche, nous allons devoir introduire un outil supplémentaire : l' *espace contraint.*

L'espace contraint Lorsque l'on considère une observation (Y_t^k) correspondant à une réalisation d'un mot donné, la durée du signal est fixé, nous la notons T. Par hypothèse, il existe un processus caché (X_t^k) à valeur dans $1 \dots N$, où N est également fixé a priori, d'après la règle empirique indiquée plus haut. Nous nous plaçons dans le cadre d'un modèle gauche-droite, c'est-à-dire que l'on passe d'un état à son successeur, sans saut. Cette dernière hypothèse s'interprète comme suit : les différents sons composant un mot sont pronnoncés les uns à la suite des autres, sans omission. Il est alors également naturel de supposer que le processus s'achève dans le dernier état : N. Le mot n'est pas tronqué. En résumé, notre processus (X_t) est initialisé au temps $t = 1$ par l'état 1 : $X_1 = 1$, et s'achève dans l'état N au temps T : $X_T = N$. Nous sommes alors confrontés à un problème : notre chaîne de Markov peut très bien ne pas avoir atteint l'état N au temps T avec une probabilité non nulle... Il suffit qu'il soit resté $T-1$ fois dans l'état 1, évènement muni de la probabilité p_1^{T-1}. Si l'on veut conserver l'ensemble des hypothèses de départ, il faut se placer dans *l'espace contraint*, la partie de $\mathcal{E} = \{(x_t) \in \{1 \dots N\}^T\}$ telle que :

$$\mathcal{E}_c = \{(x_t) \in \mathcal{E} | 1 = x_1 \leq x_2 \leq \dots \leq x_T = N\}$$

La distribution de probabilité associée En raison de nos hypothèses, nous avons été amené à nous restreindre à un espace d'états cachés plus petit que l'espace naturel $\mathcal{E}$, l'espace $\mathcal{E}_c$. Que deviennent les probabilités de nos (x_t) ? Si nous voulons conserver des propriétés intéressantes, il nous faut bien sûr conserver nos anciennes probabilités, et les renormaliser. Nous définirons $P_c((X_t) = (x_t))$ pour $(x_t) \in \mathcal{E}_c$ de la manière suivante :

$$\begin{aligned}
P_c((X_t) = (x_t)) &= P((X_t) = (x_t) | (x_t) \in \mathcal{E}_c) && (13)\\
&= \frac{P((X_t) = (x_t))}{P((X_t) \in \mathcal{E}_c)} && (14)\\
&= \frac{1}{P((X_t) \in \mathcal{E}_c)} P(X_1 = x_1) \prod_{a=2\dots t} p_{x_{a-1},x_a} && (15)\\
&= \frac{1}{P((X_t) \in \mathcal{E}_c)} \prod_{a=2\dots t} p_{x_{a-1},x_a} && (16)
\end{aligned}$$

L'écriture de $P_c((X_t) = (x_t))$ peut encore se simplifier dans le cas particulier qui nous occupe, le modèle gauche-droite. En effet, pour tout $i \in 1 \dots N$, posons $Ti = \text{Card}(t \in 1 \dots T | x_t = i)$: T_i est la durée du séjour dans l'état i. D'après nos hypothèses, comme l'on passe par tout les états, il est immédiat que

* $\forall i : 1 \leq Ti \leq T - (N-1)$
* $\sum_{i=1}^{N} T_i = T$

Munis de ces notations, réécrivons la probabilité $P_c((X_t) = (x_t))$, en notant $P((X_t) \in \mathcal{E}_c)$ P^* :

$$P_c((X_t) = (x_t)) = \frac{1}{P^*} \prod_{a=2...t} p_{x_{a-1},x_a} \tag{17}$$

$$= \frac{1}{P^*} (\prod_{i=1}^{N-1} (1-p_i)\, p_i^{T_i-1}) p_N^{T_N-1} \tag{18}$$

$$= \boxed{\frac{1}{P^*} (\prod_{i=1}^{N-1} \frac{(1-p_i)}{p_i}) \frac{1}{p_N} \prod_{i=1}^{N} p_i^{T_i}} \tag{19}$$

D'où : $P_c((X_t) = (x_t)) = \mathcal{C} \prod_{i=1}^{N} p_i^{T_i}$, avec $\mathcal{C} = \frac{1}{P^*} (\prod_{i=1}^{N-1} \frac{(1-p_i)}{p_i}) \frac{1}{p_N}$

Introduction d'un $N+1^{\text{ième}}$ état fictif Si l'on en reste à une chaîne de Markov à N états, on a vu précédemment qu'alors $p_N = 1$, ce qui signifie que le dernier état est absorbant. Cette modélisation, quoique possible, est ici inadéquate : cela revient à favoriser fortement le dernier état, qui a alors une durée moyenne nettement plus importante que celle des autres. Ceci est immédiatement visible à partir de la formule précédente :

$$\forall i : \mathrm{E}(T_i) = \sum_{(x_t) \in \mathcal{E}_c} P_c((X_t) = (x_t))\, T_i \tag{20}$$

$$= \mathcal{C} \sum_{\sum_{j=1}^{N} T_j = T, \forall j, T_j \geq 1} T_i \prod_{j=1}^{N} p_j^{T_j} \tag{21}$$

D'après l'expression précédente, $\mathrm{E}(T_i)$ est d'autant plus grand que p_i est grand, lorsque l'on fixe tous les p_j. Choisir $p_N = 1$ revient dans ces conditions à grandement favoriser cet état, et c'est pour cela qu'il faut introduire un $N+1^{\text{ième}}$ état, "fictif" au sens où il n'a pas de signification physique particulière. On consultera à ce sujet la figure 5, qui pésente les durées moyennes de chacun des états d'un modèle simplifié dont le dernier état est absorbant (plus précisément, ce modèle a trois états, et les probabilités de transition des 2 premiers états sont égales) . Autrement dit, notre $N+1^{\text{ième}}$ état "clôt" le processus : on peut imaginer qu'une dernière transition à lieu à l'instant T : on bascule de N à $N+1$. On pourrait également considérer que l'on dispose d'une chaîne de Markov au nombre d'état nettement plus grand que N, voir infini. En tout cas, on ne peut travailler avec un nombre d'états exactement égal à N.

Une remarque sur l'espace contraint Le passage à un espace des chemins plus petit, $\mathcal{E}_c$, rend caduque la propriété de stationnarité de notre processus caché (X_t). En effet, plus l'on s'approche du temps T, plus les transitions vers les états supérieurs sont probables. A tel point que si l'on est

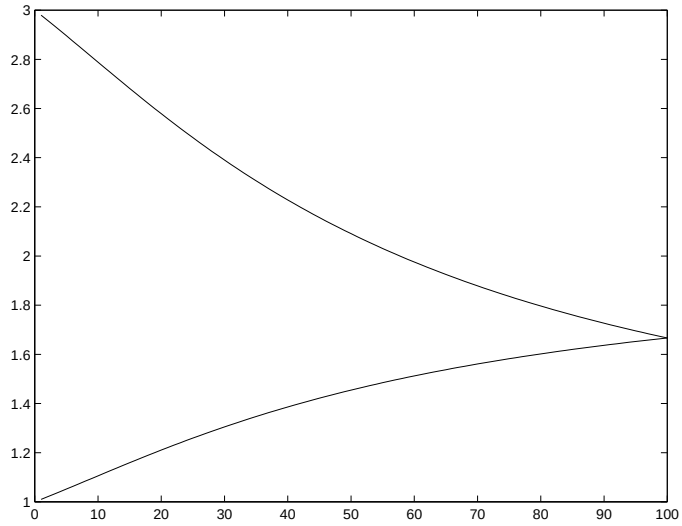

FIG. 5 – Durée moyenne des états pour $N = 3$, $T = 5$, dans le cas où 3 est absorbant. En abscisse, p varie de 0 à 100%.

dans l'état $N - n$ au temps $T - n$, les transitions vers l'état supérieur se font à coup sûr à tous les instants suivants. Du reste, le processus n'est plus Markovien du tout, même instationnaire, et ce à partir de l'instant $T - N + 1$, puisque l'espace des états possibles, E, dépend alors du temps lui aussi : on ne saurait être dans l'état 1 au temps $t - 1$. A moins que l'on ne définisse des chaînes de Markov doublement instationnaires... Mais on continura à parler de chaîne de Markov, par commodité.

2.3 Calculs dans l'espace contraint uniforme

L'espace contraint uniforme Dans cette paragraphe, nous allons nous simplifier à l'extrême notre modèle, puisque nous allons nous étudier un modèle gauche-droite "uniforme", au sens où tous les états ont une durée moyenne T_i égale. En terme de probabilités de transition, cela correspond à choisir $p_i = p_j, \ \forall i, j$. Nous noterons p cette unique probabilité. La matrice de transition est dans ce cas précis extrêmement simple :

$$\mathbb{P} = \begin{pmatrix} p & 1-p & 0 & 0 & \dots & 0 \\ 0 & p & 1-p & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & & \vdots \\ \vdots & & & & & \\ \vdots & & & \ddots & p & 1-p \\ 0 & \dots & & & 0 & 1 \end{pmatrix}$$

De la même manière, $P_c((X_t) = (x_t))$ prend une forme très simple également :

$$P_c((X_t)=(x_t)) = \frac{1}{P^*\, p_N}(\prod_{i=1}^{N-1}\frac{1-p_i}{p_i})(\prod_{i=1}^{N-1} p_i^{T_i}) \tag{22}$$

$$= \frac{1}{P^*}(1-p)^{N-1}(\frac{1}{p^N})p^{\sum_i T_i} = \frac{1}{P^*}((1-p)^{N-1}\frac{1}{p^N}p^T \tag{23}$$

$$= \frac{1}{P^*}(1-p)^{N-1}p^{T-N} \tag{24}$$

d'où on tire que $P_c((X_t) = (x_t))$ ne dépend pas de (x_t), ce qui signifie que notre "espace des chemins" est munis de la probabilité uniforme. Ce qui permet au passage d'exprimer P^* en fonction de Card$(\mathcal{E}_c)$:

$$P_c((X_t)=(x_t)) = \frac{1}{\text{Card}(\mathcal{E}_c)} = \frac{1}{P^*}(1-p)^{N-1}p^{T-N} \Longleftrightarrow \boxed{P^* = \text{Card}(\mathcal{E}_c)(1-p)^{N-1}p^{T-N}}$$

Cette formule fait penser au $N-1^{\text{ième}}$ terme d'une loi binômiale de paramètres p et $T-1$. C'est effectivement le cas, puisque nous pouvons très simplement calculer Card$(\mathcal{E}_c)$. En effet, dénombrer les éléments de cet ensemble revient à compter le nombre de possibilités existant pour placer nos $N-1$ transitions vers un état supérieur, avant l'instant T. On en deduit que Card$(\mathcal{E}_c)$ est un dénombrement :

$$\boxed{\text{Card}(\mathcal{E}_c) \;=\; C_{T-1}^{N-1}}$$

On peut alors interpréter P^* comme étant tout simplement la probabilité d'avoir $N-1$ transitions de probabilité $1-p$ sur $T-1$ tentatives, ce qui est bien donné par la loi binômiale classique !

Calculs à mener Dans la suite de ce mémoire, nous présenterons une estimation possible (approximative, mais rapide à effectuer) des paramètres de notre modèle complet de parole. Or, cette estimation repose sur le calcul de quantités statistiques associée au modèle très simple auquel est dédié ce paragraphe. Il peut sembler incongru que des calculs sur un modèle si simple puissent apporter des informations sur un modèle bien plus complexe et où les calculs sont autrement difficiles, voire impossibles, comme nous le verrons plus tard. Mais tel est bien le cas, puisqu'en fait notre modèle complet reste "proche" du modèle simpliste, dans un sens que nous discuterons plus tard.

Ces quantités s'expriment toutes en fonction de ce que nous noterons les $(D_i)_{i\in 1\ldots N+1}$, qui sont les débuts des séjours dans chaque état. Ainsi, $D_1 = 1$, et plus généralement, $\forall i :\; D_i = 1 + \sum_{j=1}^{i-1} T_i$. Utilisera la notation $D_{N+1} = T+1$, qui serait le début du $N+1^{\text{ième}}$ état. Les quantités statistiques que nous nous proposons de calculer ici sont les les suivantes :

* les expressions du type E$(T_i^A T_j^B \ldots)$, où $A,\; B \in \mathbb{N}$
* application : Var(D_i)

Dans le cadre du logiciel de reconnaissance vocale mis au point durant le stage, nous avons été aménés en fait à estimer d'autres quantités : E(D_i^2) et Var(D_i^2). Nous ne présenterons pas ici les calculs longs et fastidieux qui ont été menés à bien. Ils sont une autre application des techniques

de calcul que nous introduisons ici.

Nous allons nous appuyer sur un lemme de combinatoire, à la base de tous les calculs qui suivront :

Lemme : $\forall (N,T) \in \mathbb{N}^2$ *tels que* $N \leq T$*, on a :* $\boxed{\sum_{i=N}^{T} C_i^N \;=\; C_{T+1}^{N+1}}$

Démonstration : Cette formule, non-intuitive, est vraie en vertu d'une transformation déjà effectuée au cours des calculs : sommer sur les $(x_t) \in \mathcal{E}_c$ des quantités ne dépendant que des T_i revient à sommer ces mêmes quantités sur les $T_1, \ldots, T_N$ tels que $\sum_i T_i = T$ et $T_i \geq 1,\ \forall i$. Pour cette raison, on a : Card$(\mathcal{E}_c) = \sum_{(x_t)\in\mathcal{E}_c} 1 = \sum_{(T_i)\in\mathcal{T}(N,T)} 1$, avec $\mathcal{T}(N,T) = \{(T_1,\ldots,T_N)$ tels que $\sum_i T_i = T$ et $T_i \geq 1,\ \forall i\}$.
Il est alors aisé d'obtenir notre formule en effectuant une permutation de sommation : $\sum_{(T_i)\in\mathcal{T}(N,T)} 1 = C_{T-1}^{N-1} = \sum_{T_1=1}^{T-(N-1)} \sum_{(T_2,\ldots,T_N)\in\mathcal{T}(N-1,T-T_1)} 1 = \sum_{T_1=1}^{T-(N-1)} C_{T-T_1-1}^{N-2}$, puis en effectuant le changement de variable $a = T - T_1 - 1$: $C_{T-1}^{N-1} = \sum_{a=N-2}^{T-2} C_a^{N-2}$. D'où la formule.

Corollaire : A partir de ce résultat, par des opérations élémentaires sur les combinaisons, on a également la généralisation suivante :

$$\boxed{\forall (N,T,L) \in \mathbb{N}^3 \text{ tels que } N \leq T : \sum_{i=N}^{T} \{(\prod_{j=1}^{L}(i+j))C_i^N\} \;=\; (\frac{N+1}{N+L+1})(\prod_{j=1}^{L}(T+1+j))C_{T+1}^{N+1}}$$

L'intérêt de cette formule est qu'elle permet de calculer toutes les sommes du type $\sum_{i=N}^{T} P(i)\ C_i^T$, où $P(i)$ est une fonction polynômiale en i. Pour cela, il suffit de décomposer le polynôme selon la base des $\{\prod_{j=1}^{K}(X+j)\}_{K\in\mathbb{N}}$ et de calculer la somme comme combinaison linéaire des sommes correspondant aux éléments de cette base.

Application au calcul de E(Ti^L) Même si cela n'est pas immédiatement évident, le calcul de E(Ti^L) permet de lever la plupart des difficultés de calcul des grandeurs que nous nous proposons de calculer. On a :

$$C_{T-1}^{N-1}\mathrm{E}(Ti^L) = \sum_{(T_i)_{i=1}^N \in \mathcal{T}(N,T)} T_i^L = \sum_{T_1=1}^{T-(N-1)} T_1^L \sum_{(T_i)_{i=2}^N \in \mathcal{T}(N-1,T-T_1)} 1 \tag{25}$$

$$= \sum_{T_1=1}^{T-(N-1)} T_1^L C_{T-T_1-1}^{N-2} = \sum_{k=N-2}^{T-2} (T-(k+1))^L C_k^{N-2} \tag{26}$$

$$= \sum_{k=N-2}^{T-2} \{\sum_{i=0}^{L} C_L^i (-1)^i (k+1)^i\ T^{L-i}\} C_k^{N-2} \textit{(par application du binôme de Newton)} \tag{27}$$

$$= \sum_{i=0}^{L} C_L^i (-1)^i T^{L-i} \sum_{k=N-2}^{T-2} (k+1)^i C_k^{N-2} \tag{28}$$

En posant $(k+1)^i = \sum_{j=1}^{i} \alpha_{i,j}(k+1)\dots(k+j)$, on obtient finalement :

$$\boxed{C_{T-1}^{N-1}\mathrm{E}(Ti^L) = \sum_{i=0}^{L}\sum_{j=1}^{i} C_L^i(-1)^i T^{L-i}\alpha_{i,j}\sum_{k=N-2}^{T-2}(\prod_{a=1}^{j} k+a)C_k^{N-2}}$$

quantité que nous savons calculer en vertu du corollaire précédent (qui permet de simplifier par C_{T-1}^{N-1} !). Pour pouvoir algorithmiquement calculer cette formule (à l'aide du logiciel Matlab, par exemple), il nous faut (dernière difficulté à lever !), donner un moyen de calculer simplement les $(\alpha_{i,j})_{i,j\geq 1}$. Or, à la manière des coefficents du binôme, ces nombres se calculent également de proche en proche par la règle suivante :

$$\begin{cases} \forall i \geq 1\ ,\ \alpha_{i,i} = 1 \\ \forall i \geq 1\ ,\ \alpha_{i+1,1} = -\alpha_{i,1} \\ \forall i \geq 1\ ,\ \forall 2 \leq j \leq i\ ,\ \alpha_{i+1,j} = \alpha_{i,j-1} - j\ \alpha_{i,j} \end{cases}$$

Ce résultat se démontre par une simple récurrence, en faisant la remarque que $(k+n+1) = (k+n)+1$... Il est alors aisé de calculer numériquement les valeurs de $\mathrm{E}(Ti^L)$ pour un L donné.

Méthodologie des calculs Le calcul de $\mathrm{E}(D_i)$, $\mathrm{Var}(D_i)$, $\mathrm{E}(D_i^2)$, $\mathrm{Var}(D_i^2)$ se déduit des résultats précédents. En effet, par linéarité de l'espérance, on est ramené à des calculs du type : $\sum_{(T_i)_{i=1}^N \in \mathcal{T}(N,T)} T_1^A T_2^B T_3^C T_4^D$, avec $A+B+C+D \leq 4$. Il n'est pas nécessaire de considérer des T_i avec $i \geq 5$, puisque les T_i jouent dans notre modèle un rôle parfaitement symétrique. Nous ne présenterons pas ici l'ensemble des calculs, qui présentent de fortes similarités entre eux. A titre d'exmple, voici comment on calcule $\mathrm{E}(T_1^2T_2)$:

$$C_{T-1}^{N-1}\mathrm{E}(T_1^2T_2) = \sum_{(T_i)_{i=1}^N \in \mathcal{T}(N,T)} T_1^2T_2 \tag{29}$$

$$= \sum_{T_1=1}^{T-(N-1)} T_1^2 \sum_{(T_i)_{i=2}^N \in \mathcal{T}(N-1,T-T_1)} T_2 \tag{30}$$

$$= \sum_{T_1=1}^{T-(N-1)} T_1^2 C_{T-T_1}^{N-1} = C_T^N \mathrm{E}_{N+1,T+1}(T_1^2) \tag{31}$$

On est ainsi ramené au calcul de $\mathrm{E}_{N,T}(T_1^2)$. Au final, on trouve les formules suivantes :

$$\boxed{\begin{cases} \mathrm{E}(T_i) = \frac{T}{N} \text{ d'où } \mathrm{E}(D_i) = 1 + (i-1)\frac{T}{N} \\ Var(T_i) = \frac{T(T-N)(N-1)}{N^2(N+1)} \\ Var(D_i) = \frac{T(T-N)(i-1)(N+1-i)}{N^2(N+1)} \end{cases}}$$

En ce qui concerne des expressions telles que $Var(D_i^2)$, la formule est bien plus complexe, en raison du nombre de termes impliqués dans le calculs. Cependant, le calcul a été mené a bien, et on peut en voir le résultat à la figure suivante.

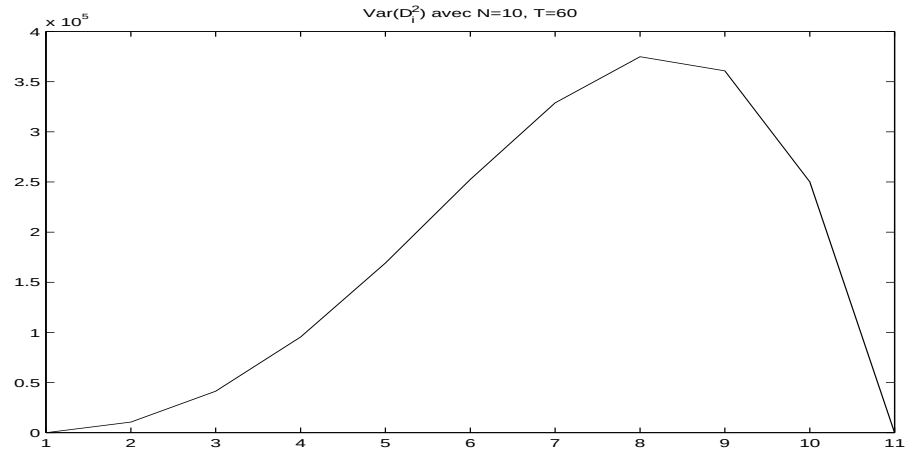

FIG. 6 – Var(D_i^2), avec i en abscisse. En ordonnées, l'échelle jusqu'à 10^5

Conclusion pour le modèle très simplifié Comme les calculs le suggèrent, il est possible dans le cadre du modèle gauche-droite uniforme dans l'espace contraint de caculer les espérances de n'importe quelle variable aléatoire donnée comme fonction algébrique des T_i. Ce fait est remarquable, et il nous permettra d'effectuer une approximation très rapide des paramètres de notre modèle complet de parole. Malheureusement, le recours aux techniques combinatoires que nous venons de présenter n'est plus possible sitôt que l'hypothèse d'uniformité est mise en défaut, ce qui sera le cas en général ! Pratiquement, quasiment aucune de ces grandeurs ne sera calculable exactement, et seules des techniques d'estimation de type "Monte-Carlo" (estimations par la simulation), bien plus coûteuses en temps de calcul que ne le serait un hypothétique calcul exact, sont envisageables. Tel est le prix à payer pour une plus grande complexité du modèle, indispensable pour de bonnes performances en reconnaissance vocale.

3 Les champs de Markov : un outil pour la modélisation multibande du signal de parole

Nous introduisons dans cette partie l'outil qui fait l'originalité de notre étude de la parole : les *champs de Markov*. En effet, actuellement, les HMM sont presque le seul outil utilisé pour la reconnaissance vocale. Cette approche s'inscrit dans le prolongement des travaux de thèse de Guillaume Gravier (voir à ce sujet [6]). Une des originalités de ce travail consiste à utiliser comme "variables réduites" les D_i^k, qui sont une représentation bien plus compacte des segmentations cachées que les X_t^k. C'est par rapport à ces variables que l'on formulera certaines énergies d'interaction.

3.1 Les origines des champs de Markov en Physique Statistique

Avant de définir mathématiquement les Champs de Markov, arrêtons-nous quelques instants sur leurs origines historiques. En effet, cet outil mathématique a été inventé pour les besoins d'une discipline qui s'est formidablement développé au 20ème siècle : la Physique Statistique. Nous allons commencer par rappeller brièvement les axiomes de cette théorie scientifiques, et nous verrons ensuite comment la nécessité d'un nouvel outil a pu apparaître. Le lecteur pourra se référer à [1] pour plus de détails.

Les axiomes de la Physique Statistique Nous allons nous placer dans le cadre de ce que l'on appelle le *modèle canonique* : l'objet d'étude de la physique statistique est alors un système de N particules interagissant entre elles et avec le milieu extérieur d'une façon que nous ne préciserons pas pour l'instant. On fait l'hypothèse de la *quantification* des niveaux énergétiques que peut occuper notre système : les énergies forment une suite croissante (E_n) finie ou non, mais où le nombre d'état est en tout cas très supérieur à 1 (rappellons pour mémoire que le nombre N de particules sera en général astronomiquement grand, de l'ordre pour un système macroscopique du nombre d'Avogadro, soit à peu près 10^{23} ! Le nombre d'état est alors au moins de l'ordre de N^2 si ce n'est beaucoup plus... Il n'y a alors plus grande différence avec un nombre d'états infini...).
La description que l'on fait du système est de type *probabiliste* : avoir une parfaite connaissance du système à un instant t signifiera que l'on connait exactement l'ensemble $\{p_n\}$ des probabilités qu'a le système de se trouver dans chacun des états d'énergie E_n. Comment obtient-on les probabilités p_n en fonction des E_n ? La réponse apportée par la Physique Statistique repose sur la maximisation d'une grandeur nommée *entropie statistique*, notée $H((p_n))$. Plus précisément lorsque le système a une énergie moyenne U, les probabilités (p_n) sont données par l'unique solution du problème de maximisation contrainte suivant :

$$\begin{cases} (p_n) = arg \ \max\limits_{(p_n)} \ S((p_n)), \text{ ou } S((p_n)) = -\sum_n p_n \log(p_n) \\ \text{et ceci sous les contraintes :} \\ \\ \sum_n p_n = 1 \text{ \textit{(les probabilités sont normalisées à 1)}} \\ \sum_n E_n \, p_n \ = \ U \text{ \textit{(énergie moyenne égale à } } U \text{ \textit{)}} \end{cases}$$

Ce problème de maximisation admet une unique solution (au moins en dimension finie !), que l'on peut calculer à l'aide de la technique des multiplicateurs de Lagrange. Le résultat important

est que l'on obtient le résultat suivant, à savoir que le comportement du système est système est régis par la statistique de Gibbs-Boltzmann :

$$\forall E \in (E_n) \ : \ P(\text{Energie } = E) = \frac{e^{-\beta(E - S(E)\ T)}}{Z}$$

avec $\beta = \frac{1}{kT}$ (β est le multiplicateur de Lagrange associé à U), où k est le nombre de Boltzmann, T la température du système (qui déterminée par l'énergie moyenne de celui-ci). $S(E) = k\ \log(N(E))$ est l'entropie associée à l'énergie E, $N(E)$ étant le degré de *dégénérescence* de l'énergie E, id est le nombre d'états possibles n d'énergie E. Enfin, et ceci est une notion très importante : Z est la *fonction de partition* associée à notre distribution de Gibbs-Boltzmann : c'est sa constante de normalisation à l'unité, donnée par :

$$Z = \sum_{E \in (E_n)} e^{-\beta(E - S(E)\ T)}$$

Un des problèmes les plus délicats en Physique Statistique réside justement dans le calcul de cette quantité, souvent extrêmement malaisé, même pour un nombre d'énergie fini. La difficulté réside dans la grandeur du cardinal de l'ensemble sur lequel s'effectue la sommation : il est en général extraordinairement grand, et il ne serait être question de calculer cette somme en agrégeant les termes un à un... Nous reviendrons vite à ce problème, qui concerne également la reconnaissance vocale par champs markoviens !

Remarque sur le rôle de la température Revenons un instant sur l'expression de la probabilité $P(\text{Energie } = E)$, et intéressons-nous à l'influence de la température sur celle-ci. Lorsque la température du système (grandeur toujours positive en physique, du moins sur l'échelle de Kelvin), tend vers 0, on une propriété physique très intéressante : seuls les états d'énergie la plus basse ont une probablité non-nulle d'être mesurés. Au contraire, lorsque T tend vers $+\infty$, les états deviennent équiprobables : le désordre règne dans le système. Nous démontrerons ces propriétés dans le cadre des champs markoviens un peu plus loin (pour des espaces finis de configurations)

Un exemple célèbre : le modèle d'Ising Nous allons voir apparaitre naturellement la notion de champ markovien en étudiant un modèle simple et classique en physique : le modèle d'Ising. Décrivons-le brièvement :

- le système est composé de N particules, fixes sur les points d'un quadrillage régulier dans un espace à 2 dimensions
- ses particules n'interagissent entre elles que par leurs moments magnétiques $(S_n)_{n \in 1 \ldots N}$ (appelés aussi "spins"), à valeur dans $\{-1, 1\}$
- leur énergie d'interaction mutuelle est la suivante :
  $U_{spin,\ spin}((S_n)) = -\sum_{(i,j) \in \mathcal{C}} E_{i,j}\ S_i\ S_j$, où $\mathcal{C}$ est l'ensemble des couples "voisins" : l'idée est que l'on ne retient les interaction qu'*au voisinage* de chacun des sites du réseau. pour des systèmes *ferromagnétiques*, $E_{i,j} \geq 0$: ce coefficient est d'autant plus grand que le couplage des spins S_i et S_j est fort.
- le système est soumis à un champ magnétique exterme (homogène ou non) (B_n) qui s'applique en chacun des points du réseau. L'énergie d'interaction correspondante est la suivante : $U_{champ,\ spin} = -\sum_n S_n\ B_n$.

Ainsi l'énergie du système pour une configuration de spins donnée s'écrit :

$$E = -\sum_{(i,j)\in\mathcal{C}} E_{i,j}\, S_i\, S_j - \sum_n S_n\, B_n$$

et la distribution de probabilités sur les états est donnée par :

$$P((S_n) = (s_n)) = \frac{e^{-\beta E((s_n))}}{Z}, \text{ avec } Z = \sum_{(s'_n)\in\{-1,1\}^N} e^{-\beta E((s'_n))}$$

Les configurations les plus probables étant celles qui minimisent l'énergie totale du système, on voit ici qu'il y a compétition entre deux phénomènes : l'alignement des spins 2 à 2 au seins des différents voisinages, et l'alignement spin par spin avec le champ magnétique extérieur. L'énergie prend ici une expression complexe, et il n'est pas évident de déterminer la configuration d'énergie minimale.

Le point de vue probabiliste On peut très bien considérer les spins du système comme autant de variables aléatoires S_n à valeur dans $\{-1, 1\}$. Les lois des S_n se déduisent de la probabilité conjointe $P((S_n) = (s_n))$. Ces variables aléatoires ont une propriété remarquable que nous démontrerons dans le cadre général des champs de markov :

$$P(S_n = s_n|(S_m)_{m\neq n} = (s_m)_{m\neq n}) = P(S_n = s_n|(S_m)_{(m,n)\in\mathcal{C}} = (s_m)_{(m,n)\in\mathcal{C}})$$

Autrement dit, la valeur prise par un spin ne dépend que des valeurs des autres spins à son voisinage si l'on connait les valeurs prises par le champ aux autres sites du système. Il n'y a donc pas de dépendance à longue distance. Cette propriété généralise la propriété dite de "Markov faible" que nous avions vu au chapitre précédent. On voit ainsi apparaître une propriété plus générale, qui servira de définition à une classe d'objets plus vaste encore que les chaînes de Markov : les champs de Markov. Le paragraphe suivant leur est dédié.

3.2 Définition et propriétés générales

Définition d'un champ Markovien Soit E un espace d'*états*, S un ensemble de *sites* que l'on suppose dénombrables. Soit $\mathfrak{V}$ un ensemble de *voisinages* pour $S : \mathfrak{V} :\ S \to \mathcal{P}(S)$, tel que (1) : $\forall s \in S, s \in \mathfrak{V}_s$, **(2)** : $\forall (s_1, s_2) \in S^2,\ (s_2 \in \mathfrak{V}_{s_1}) \iff (s_1 \in \mathfrak{V}_{s_2})$, et (3) : $\exists M |\, \forall s,\ \text{Card}(\mathfrak{V}_s) < M < \infty$. Un champ markovien sur S a valeur dans E est une famille $(X_s)_{s\in S}$ de variables aléatoires à valeur dans E vérifiant la propriété suivante :

$$\boxed{\forall s \in S,\ \forall (x_r) \in E^S :\ P(X_s = x_s|(X_r)_{r\neq s} = (x_r)_{r\neq s}) = P(X_s = x_s|(X_r)_{r\in\mathfrak{V}_s\setminus s} = (x_r)_{r\in\mathfrak{V}_s\setminus s})}$$

Cette propriété signifie simplement que la valeur de la variable aléatoire X_s, conditionnellement à la connaissance du champ ailleurs qu'en s, ne dépend que des valeurs prises par le champ à son voisinage.

Exemple Lorsque le voisinage d'un site est trivialement réduit à lui-même, n'importe quel champ de Markov associé est simplement un champ de variables aléatoires indépendantes.

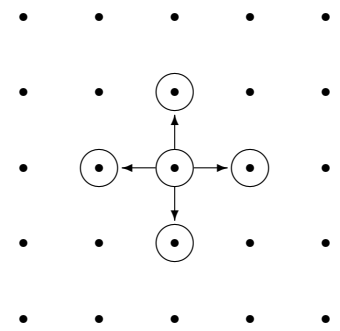

FIG. 7 – Voisinage dit "4-connexe" dans un réseau à maille rectangulaire

Un concept clé : les cliques Une notion intéressante est associée au concept de voisinage : la notion de *clique* d'un *ordre* donné. En effet, sous les mêmes hypothèses que lors de la définition précédente, on définit l'ensemble $\mathcal{C}_k,\ k \in \mathbb{N}$ des cliques d'ordres k comme suit : $\mathcal{C}_k = \{(s_i)_{i\in 1\ldots k} \in S^k | \forall i \in 1\ldots k,\ \forall j \in 1\ldots k,\ j \neq i,\ s_j \in \mathfrak{V}_{s_i}$ et $s_i \neq s_j\}$. Il s'agit donc des sous-ensembles de sites qui sont mutuellement au voisinage les uns des autres. L'hypothèse (2) de la définition précédente assure qu'il existe un certain ordre R au-delà duquel il n'existe plus de cliques : il suffit pour cela de prendre $R = M$. Les cliques les plus couramment utilisées sont les cliques d'ordre 1 (on a bien sûr $\mathcal{C}_1 = S$), et les cliques d'ordre 2. Dans l'exemple du modèle d'Ising présenté au paragraphe précédent, seules ces deux types de cliques intervenaient.

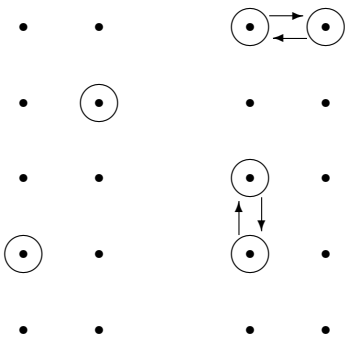

FIG. 8 – Les 2 seuls types de cliques en 4-connexité : ordre 1 (à gauche) et ordre 2 (à droite)

Les champs de Gibbs Introduisons un deuxième type de champ, en nous inspirant de la Physique Statistique. En gardant les mêmes hypothèses que précédemment, un champ de Gibbs sur S à valeur dans E, noté (X_s), est un champ de variables aléatoires tel que $\exists U : S^E \to \mathbb{R}$, appelée *fonction énergie*, pouvant se décomposer en $U((x_s)) = \sum_{k=1}^{M} U_k((x_s))$, avec $\forall k :\ U_k((x_s)) =$

$\sum_{(s_1,\ldots,s_k)\in\mathcal{C}_k} E_c^{(s_1,\ldots,s_k)}(x_{s_1},\ldots,x_{s_k})$. La probabilité pour le champ (X_t) de se réaliser en (x_t) est alors donnée par :

$$P((X_t)=(x_t)) = \frac{e^{-U((x_t))}}{Z} \text{ avec } Z = \sum_{(x_a)} e^{-U((x_a))}$$

comme auparavant, Z est appelée *fonction de partition*. on voit ici que les contribution à l'énergie d'une configuration donnée ont plusieurs origines a priori : une par type de clique. Ainsi, dans le modèle d'Ising, l'énergie du système comporte une contribution du fait de l'alignement des spins sur le champ magnétique extérieur (cliques d'ordre 1) et une autre dûe à l'interaction spin-spin (cliques d'ordre 2). Pour des espaces finis, la configuration la plus probable, qui est *celle qui réalise le mimimum de l'énergie*, résulte d'une minimisation équilibrée entre toutes les contributions à l'énergie, ce qui peut rendre sa recherche pratique extrêmement ardue, eu égard au cardinal très élevé de l'espace de toutes les réalisations possibles (pour l'espace des spins : 2^N, avec $N \simeq 10^{23}$!).

Le théorème de Hammersley-Clifford Voici une propriété fondamentale des champs de Markov, qui fait le lien avec les champs de Gibbs. Il s'énonce de la manière suivante :
avec les hypothèses spécifiées en début de chapitre (espaces dénombrables, voisinages bornés) :

1. *tout champ de Gibbs est un champ de Markov*
2. *tout champ de Markov n'ayant aucune configuration de probabilité nulle est également un champ de Gibbs*

Ainsi, dans l'hypothèse où toutes les configurations peuvent se réaliser selon une probabilité non nulle, il y a équivalence entre champs de Markov et de Gibbs.

Démonstration Nous nous contenterons de donner ici la preuve de la première assertion, la deuxième étant bien plus difficile à montrer. Soit (X_s) un champ de Gibbs. Calculons $P(X_s = x_s|(X_r)_{r\neq s} = (x_r)_{r\neq s})$, en notant $C_k(s)$ l'ensemble des cliques d'ordre k auxquelles le site s n'appartient pas :

$$\mathcal{P} = P(X_s = x_s|(X_r)_{r\neq s} = (x_r)_{r\neq s}) = \frac{P((X_a)=(x_a))}{P((X_r)_{r\neq s}=(x_r)_{r\neq s})} \tag{32}$$

$$= \frac{P((X_a)=(x_a))}{\sum_{x\in E} P((X_r)_{r\neq s}=(x_r)_{r\neq s}\ ,\ X_s = x)} \tag{33}$$

$$= \frac{\frac{e^{-U((x_a))}}{Z}}{\sum_x \frac{e^{-U((x_1,\ldots,x,\ldots x_{|E|}))}}{Z}} \tag{34}$$

$$= \frac{e^{-\sum_k U_k((x_a))}}{\sum_x e^{-\sum_k U_k((x_1,\ldots,x,\ldots x_{|E|}))}} \tag{35}$$

$$= \frac{\prod_k e^{-\sum_{c\in C_k(s)} E_k^c((x_c^i)_{i\in 1\ldots k}) - \sum_{c\notin C_k(s)} E_k^c((x_c^i))}}{\sum_x \prod_k e^{-\sum_{c\in C_k(s)} E_k^c((y_c^i)) - \sum_{c\notin C_k(s)} E_k^c((y_c^i))}} \tag{36}$$

Où les x_c^i et les y_c^i sont obtenus respectivement en extrayant selon la clique c les valeurs correspondant dans (x_a) et $^(x_1, \ldots, x, \ldots, x_{|E|})$. Séparons notre expression en 2 parties, l'une dépendant de s et l'autre non :

$$\mathcal{P} = \frac{\prod_k e^{-\sum_{c \in C_k(s)} E_k^c((x_c^i)_{i \in 1 \ldots k})}}{\prod_k e^{-\sum_{c \in C_k(s)} E_k^c((x_c^i)_{i \in 1 \ldots k})}} \frac{\prod_k e^{-\sum_{c \notin C_k(s)} E_k^c((x_c^i))}}{\sum_x \prod_k e^{-\sum_{c \notin C_k(s)} E_k^c((y_c^i))}} \tag{37}$$

$$= \frac{\prod_k e^{-\sum_{c \notin C_k(s)} E_k^c((x_c^i))}}{\sum_x \prod_k e^{-\sum_{c \notin C_k(s)} E_k^c((y_c^i))}} \tag{38}$$

$$= \frac{P((X_r)_{r \in \mathfrak{V}_s \setminus s} = (x_r)_{r \in \mathfrak{V}_s \setminus s}, X_s = x_s)}{P((X_r)_{r \in \mathfrak{V}_s \setminus s} = (x_r)_{r \in \mathfrak{V}_s \setminus s})} \tag{39}$$

$$= P(X_s = x_s | (X_r)_{r \in \mathfrak{V}_s \setminus s} = (x_r)_{r \in \mathfrak{V}_s \setminus s}) \tag{40}$$

D'où le résultat.

Simulation des Champs de Gibbs A priori, la forme générale prise par les lois de probabilité des champs de Gibbs rend leur manipulation difficile, notamment en raison de la fonction de partition, qu'il est la plupart du temps impossible de calculer. Cela pose toutes sortes de problèmes, car on ne peut pas calculer non plus de très nombreuses grandeurs statistiques associées au champ : la loi marginale d'un X_s est en général inconnue, par exemple. Ce problème peut en pratique être contourné grâce à un résultat très important : *il est possible de simuler numériquement* les champs de Gibbs (et donc les champs de Markov auxquels nous nous intéresserons). Cela a pour conséquence qu'il est possible d'estimer toutes les grandeurs statistiques associées au champ avec une précision arbitraire, ne dépendant que du temps de calcul que l'on veut bien y consacrer. La loi de X_s devient accessible par ce biais ! Ce type de technique fait partie de ce que l'on appelle les algorithmes de *Monte-Carlo.*
Nous ne présenterons pas ici la théorie complète de la simulation des champs de Gibbs, qui nécessiterait de copieux développements, que l'on pourra trouver dans [3]. Nous nous contentons ici de présenter l'algorithme de *Metropolis*, tel qu'il a été implémenté informatiquement dans notre logiciel de reconnaissance vocale.

L'algorithme de Metropolis Cet algorithme est appelé un *échantilloneur*, puisqu'il produit des échantillons de notre champ de Gibbs (c'est-à-dire qui suivent sa loi de probabilité). Evidemment, l'échantillonage exact ne se produit qu'asymptotiquement : les premiers échantillons sont inutilisables, et il faut attendre plusieurs dizaines, voire centaines, d'itérations pour pouvoir les utiliser. Voici comment l'on procède :

1. on part d'une configuration initiale (x_s)
2. on parcourt dans un ordre arbitraire l'ensemble des sites, S(fini), et à chaque nouveau site, on tire un nouvel état possible avec la probabilité uniforme sur E (supposé fini). On accepte ou l'on rejette cette modification selon la règle suivante :

$$\begin{cases} (1) \text{ si la variation d'énergie } \Delta U \text{ occasionnée est } < 0, \text{ alors la modification est conservée} \\ (2) \text{ dans le cas contraire elle est acceptée avec la probabilité } p = e^{-\Delta U}. \end{cases}$$

3. on boucle sur l'étape numéro 2, autant de fois que nécessaire.

Cet algorithme offre d'excellentes performances en pratique pour les raisons suivantes :

- il ne fait intervenir que l'énergie du système, et absolument pas la fonction de partition
- la différence ΔU d'énergie ne fait intervenir des calculs qu'au voisinage du site s concerné, grâce à la forme très particulière de l'énergie pour les champs de Gibbs.

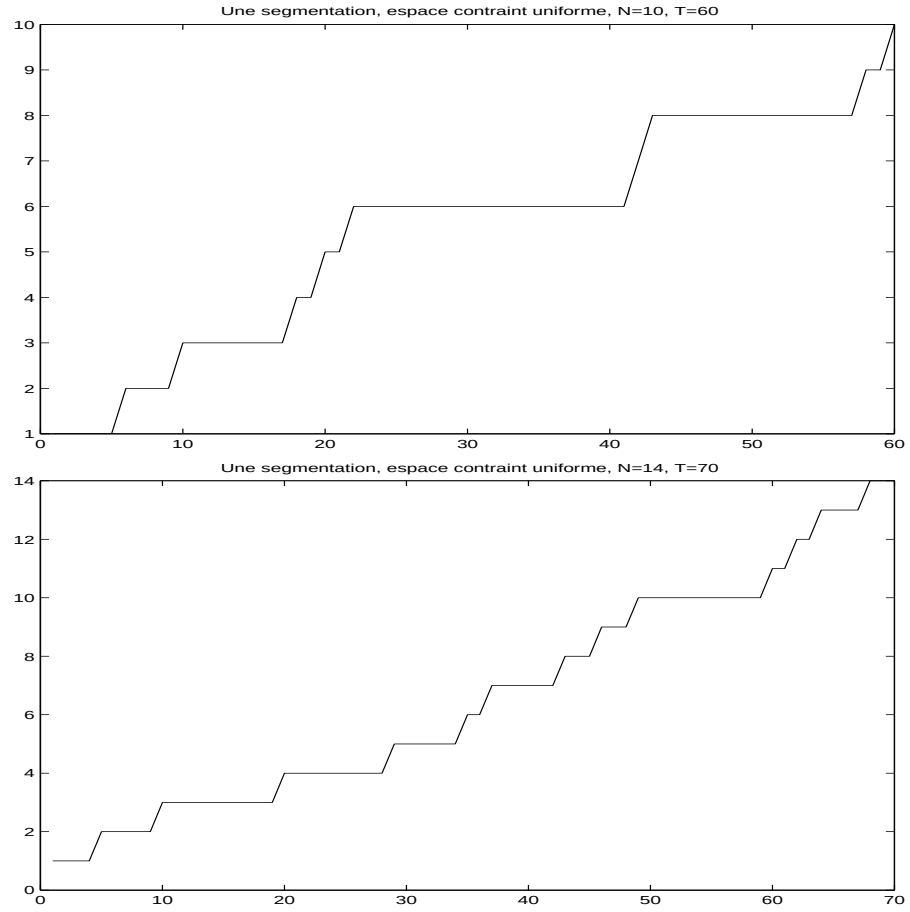

FIG. 9 – Application : deux échantillons de modèle caché dans l'espace contraint, pour la loi uniforme

Configurations d'énergie minimale Lorsque l'on manipule des champs de Markov, l'un des problèmes cruciaux que l'on rencontre est l'obtention de la (ou des) configuration(s) d'énergie minimale, par exemple afin de pouvoir appliquer l'approximation de Viterbi présentée dans la première partie de ce mémoire. Paradoxalement, le problème est difficile à résoudre, même lorsque l'on travaille sur des ensembles finis (ce sera notre cas). La difficulté est simple : l'ensemble des configurations étant gigantesque, il est impossible de le parcourir entièrement pour chercher les configurations d'énergie minimale. Une méthode envisageable consisterait à parcourir l'ensemble des sites s et à chaque fois de choisir pour nouvelle valeur de x_s celle qui minimise l'énergie globale, les autres x_a restant inchangés. On aboutirait intérativement à un minimum de l'énergie,

mais malheureusement, celui-ci ne serait que local, à moins d'un hasard extraordinaire. Dans ces conditions, comment obtenir le minimum global ? Une réponse a été apportée à ce problème difficile : c'est l'*algorithme du recuit simulé*. Mais d'y venir, présentons une notion importante : la *température* d'un champ de Gibbs, et présentons ses propriétés principales.

Notion de température Comme en Physique Statistique, il est possible de faire intervenir un paramètre β arbitraire (égal à $\frac{1}{k_B T}$ en physique). A partir d'un champ de Gibbs donné, (X_s), on peut ainsi définir une famille de champs de Gibbs (X_s^β), dont l'énergie s'exprime comme suit :

$$\forall k\ ,\ \forall (x_s):\ U_k^\beta(x_s) = \beta\, U_k(x_s)$$

On dira que $\frac{1}{\beta}$ est la *température du champ* (X_s^β) relativement au champ (X_s) (mais on pourra très bien omettre la référence à (X_s) si le contexte ne prête pas à confusion). La proposition suivante résume les propriétés les plus intéressantes liées à la notion de température :

Proposition *Supposons que E et S soient finis. Soit (X_t) un champ de Gibbs à valeur dans E. Notons $\mathcal{E}_{min}$ et $\mathcal{E}_{max}$ respectivement les ensembles des réalisations d'énergie mimimale et maximale. On notera également $\mathcal{E}$ l'ensemble des configurations possibles. On a :*

1. $P((X_s^\beta) = (x_s)) \underset{\beta\to 0}{\longrightarrow} \frac{1}{Card(\mathcal{E})}$
2. $P((X_s^\beta) = (x_s)) \underset{\beta\to +\infty}{\longrightarrow} \frac{1}{Card(\mathcal{E}_{min})}\, \mathbf{1}_{(x_s)\in\mathcal{E}_{min}}$ *où :* $\begin{cases} \mathbf{1}_{(x_s)\in\mathcal{E}_{min}} = 1 \text{ si } (x_s)\in\mathcal{E}_{min} \\ \mathbf{1}_{(x_s)\in\mathcal{E}_{min}} = 0 \text{ sinon.} \end{cases}$
3. $P((X_s^\beta) = (x_s)) \underset{\beta\to -\infty}{\longrightarrow} \frac{1}{Card(\mathcal{E}_{max})}\, \mathbf{1}_{(x_s)\in\mathcal{E}_{max}}$

Ainsi, lorsque la température tend vers $+\infty$, les configurations deviennent équiprobables : le désordre du système est complet, au sens où son entropie devient maximale. Par contre, lorsque la température tend vers 0^+, seules les configurations les plus probables (d'énergie dite fondamentale en physique) peuvent se réaliser : on tend vers les états d'énergie minimale. Enfin, si la "température" tend vers 0^- (ce qui n'a pas de sens physiquement parlant), seules les configurations d'énergie maximale peuvent se réaliser (on travaille en fait avec le champs de Gibbs dont l'énergie est l'opposé de la précédente, et où les niveaux d'énergie les plus bas sont les niveaux les plus énergétiques du précédent, par changement de signe).

Démonstration Par hypothèse, les ensembles considérés sont finis. Le problème est ici une simple question de passage à la limite. Si l'on note U_{min} l'énergie minimale du système, on a : $P((X_s^\beta) = (x_s)) = \frac{e^{-\beta U((x_s))}}{Z} = \frac{e^{-\beta U_{min} - \beta(U((x_s)) - U_{min})}}{e^{-\beta U_{min}} \sum_{(x_a)} e^{-\beta(U((x_a)) - U_{min})}} = \frac{e^{-\beta(U((x_s)) - U_{min})}}{\sum_{(x_a)} e^{-\beta(U((x_a)) - U_{min})}}$.
Lorsque β tend vers $+\infty$, tous les $e^{-\beta(U((x_a)) - U_{min})}$ tend vers 0 quand $U((x_a)) > U_{min}$. D'où la formule lorsque $\beta \longrightarrow +\infty$. On procède exactement de la même manière dans les deux autres cas de figure.

Le recuit simulé Pour obtenir une configuration d'énergie minimale, l'idéal serait de pouvoir échantillonner notre champ de Gibbs à tempérture nulle... ce qui n'est pas possible que cela correspond à $\beta = +\infty$! Si cela n'est pas réalisable stricto sensu, il est néanmoins possible d'y parvenir asymptotiquement, d'après les résultats sur la convergence de l'algorithme.
En voici les étapes :

1. on part une température T_0, suffisament "haute"
2. on échantillonne notre champ pour la température T_0, par exemple à l'aide de l'algorithme de Metropolis. Cette phase conclut l'initialisation de notre algorithme
3. une fois l'algorithme initialisé, on passe d'une température T_N à T_{N+1} en la faisant diminuer selon une règle donnée
4. on échatillonne à nouveau notre champ, cette fois à la température T_{N+1}, à partir de la configuration obtenu par échantillonage à la T_N, ce qui clôt l'itération.
5. on itère jusqu'à être "proche" de zéro.

L'algorithme a été baptisé "recuit" parce qu'il simule un procédé courant en métallurgie : on abaisse brutalement la température d'un acier que l'on vient tout juste de produire afin de lui faire atteindre un état de grande stabilité lui conférant une grande résistance aux chocs.
D'après [3], il suffit de choisir une suite de T_N proportionnelle à $\frac{1}{\log(N)}$ pour que la convergence de l'algorithme soit garantie. Cette décroissante étant fort lente, on adopte en pratique une décroissance géométrique, qui permet, toujours d'après [3] d'obtenir de très bons résultats, c'est-à-dire une configuration d'énergie quasiment minimale. On dispose ainsi grâce au recuit simulé d'un procédé permettant de mettre en œuvre l'approximation de Viterbi.

3.3 Application à la modélisation de la parole

A présent que nous avons introduit la plupart des outils dont nous avons besoins pour modéliser la parole, nous pouvons passer à la présentation du modèle markovien complet. L'objet de cette modélisation est un ensemble des mots w formant un dictionnaire W. Ces mots sont supposés *isolés*, et on va chercher à les caractériser par un certain nombre de paramètres. Comme nous l'avons précisé auparavant, notre modélisation par processus caché fait intervenir les variables aléatoires (X_t^k), qui correspondent à l'état dans lequel le système se trouve à l'instant t dans la bande k. Nous nous plaçons dans l'espace contraint $\mathcal{E}_c^K$, où K est le nombre de bandes retenu (ici, 24).

Lien entre champs en chaînes de Markov En guise de préambule, montrons comment les chaînes de Markov, et donc la modélisation par processus cachés introduite dans la 2ème partie, s'intégre au formalisme des champs de Markov. On a vu dans la deuxième partie de ce mémoire que pour une bande donné k, en supposant les bandes indépendantes, on a :

$$P_c((X_t^k)_t = (x_t^k)_t) \;=\; \mathcal{C} \prod_{i=1}^{N} (p_i^k)^{T_i^k} \text{ , avec } \mathcal{C} = \frac{1}{P^{*,k}} \Big(\prod_{i=1}^{N-1} \frac{(1-p_i^k)}{p_i^k} \Big) \frac{1}{p_N^k}$$

Cela s'écrit encore :

$$\begin{aligned} P_c((X_t^k)_t = (x_t^k)_t) \;&=\; \mathcal{C} \prod_{i=1}^{N} e^{-(-T_i^k \log(p_i^k))} && (41) \\ &=\; \mathcal{C} \prod_{i=1}^{N} e^{-(T_i^k \alpha_i^k)} \text{ avec } \alpha_i^k = -\log(p_i^k) && (42) \\ &=\; \mathcal{C}\, e^{-(\sum_i A_i^k\, D_i^k)} \text{ avec } A_i^k = \alpha_{i-1}^k - \alpha_i^k \text{ en posant } \alpha_0^k = 0 && (43) \end{aligned}$$

D'où la formule générale :

$$P_c((X_t^k)_t = (x_t^k)_t) \ = \ \frac{e^{-(\sum_i A_i^k \ D_i^k)}}{Z}$$

On voit ici apparaître une formulation en terme de champ de Gibbs, mais part rapport aux variables aléatoires D_i^k, qui sont les débuts des séjours dans chacun des états dans les bandes de fréquences. C'est pour cette raison que nous raisonnerons par la suite par rapport aux D_i^k, et non par rapport aux (X_i^k). Les (D_i^k) véhiculent la même information que les (X_i^k) dans le cadre de notre modèle contraint, mais ce de manière bien plus compacte. Par rapport à ces variables, le modèle des HMM contraints indépendemment dans chaque bande est un champ de Markov d'ordre 1. Son énergie s'écrit :

$$\boxed{U_{HMM}((D_i^k)_{i,k}) = \sum_{i,k} A_i^k \ D_i^k}$$

On voit que plus A_i^k est grand, plus D_i^k tendra à être petit dans la configuration d'énergie minimale : ce sont les états postérieurs à i qui sont favorisés. Plus A_i^k sera petit (et même négatif), plus longs seront les états antérieurs à i.
On remarquera ici que'il existe une redondance entre les T_i : leur somme est égale à T. Cela a pour conséquence que l'on peu prendre $A_1^k = 0$, puisque $D_1^k = 1$ quoi qu'il arrive ! Ainsi, dans note étude, tous les coefficients correspondant au premier état seront pris égaux à 0. Tout ceci est une conséquence directe du choix de l'espace contraint. Au passage, on remarquera également qu'à une même série de A_i^k correspondent une infinité de valeurs possibles pour les p_i^k ! Ce résultat paradoxal ne l'est pas tant que cela, car pour l'espace contraint uniforme, on voit bien que rien ne dépend de la valeur de p. De même, ici, *les p_i^k seront définis à une constante multiplicative près*, ce qui n'a rien de gênant en pratique.

Modélisation de la synchronie entre bandes de fréquences La grande originalité de notre travail sur la modélisation de la parole consiste à faire intervenir une modélisation de la synchronie ou de l'asynchronie existant entre bandes de fréquences. Plus précisément, nous allons introduire une énergie complémentaire mesurant la désynchronisation entre bandes de fréquences. Physiquement, cela signifie que nous allons tenter de caractériser un mot non seulement par la durée de ses différents états, mais aussi par les synchronisations ou désynchronisations qui existent entre bandes de fréquences. Par synchronisation, on entend simultanéité des débuts du séjour dans un même état dans 2 bandes différentes. Dans la pratique, on l'observe forcément, puisque les caractéristiques du son changent dans toutes les bandes de fréquences à peu près en même temps lorsque l'on passe d'un son à un autre.
L'énergie de synchronisation que nous proposons est la suivante :

$$\boxed{U_{synchro}((D_i^k) = \sum_{i,k,l} F_i^{k,l} |D_i^k - D_i^l|^\beta \text{ avec } \beta \in \mathbb{R}_+^*}$$

Cette énergie modélise bien la synchronisation des débuts : si $F_i^{k,l}$ est grand, les débuts D_i^k et D_i^k seront proches, au sens des configurations les plus probables.

Le modèle caché complet Pour obtenir le modèle markovien caché complet, il suffit d'agréger les 2 contributions énergétiques. Cela signifie qu'à la manière du modèle d'Ising, on favorise d'une part dans chacune des bandes certains états par rapport à d'autres, et que d'autre part, on force plus ou moins la synchronie entre bandes, et ceci spécifiquement à chaque état et chaque couple de bandes.
Dans ce cadre, la distribution de probabilités de notre modèle complet est donnée par :

$$P((D_i^k)_{i,k} = (d_i^k)_{i,k}) \;=\; \frac{e^{-U_{HMM}((d_i^k))-U_{synchro}((d_i^k))}}{Z^{\,prior,\,m}_{\underline{A}\,,\,\underline{F}}} \text{ où } Z^{\,prior,\,m}_{\underline{A}\,,\,\underline{F}} \text{ est la fonction de partition associée.}$$

$$\text{avec } \underline{A} = (A_i^k) \text{ et } \underline{F} = (F_i^{k,l})$$

Notations On appelera *potentiel* associé à l'énergie $U_{HMM}((D_i^k)_{i,k})$ la fonction vectorielle $\underline{\Psi}_{\,\underline{A}}$, définie par : $\underline{\Psi}_{\,\underline{A}}((D_i^k)) = (D_i^k)_{i,k}$ et de la même manière le potentiel associé à l'énergie $U_{synchro}((D_i^k)_{i,k})$, $\underline{\Psi}_{\,\underline{F}}$, sera tel que $\underline{\Psi}_{\,\underline{F}}((D_i^k)) = (|D_i^k - D_i^l|^\beta)_{i,k,l}$, afin que l'on ait :

$$U_{HMM} = \underline{\Psi}_{\,\underline{A}} \,.\, \underline{A} \text{ et } U_{synchro} = \underline{\Psi}_{\,\underline{F}} \,.\, \underline{F} \text{ au sens du produit scalaire naturel.}$$

On peut alors réécrire notre distribution de probabilités comme suit :

$$\boxed{P((D_i^k)_{i,k} = (d_i^k)_{i,k}) = \frac{e^{-\underline{\Psi}_{\,\underline{A}} \,\cdot\, \underline{A} - \underline{\Psi}_{\,\underline{F}} \,\cdot\, \underline{F}}}{Z^{\,prior,\,m}_{\underline{A}\,,\,\underline{F}}}}$$

Modélisation de l'observation (Y_t^k) La modélisation retenue ici pour le signal est presque la plus simple envisageable, puisque nous allons considéré que lorsque notre système est dans l'état i dans la bande k, notre observation Y_t^k va suivre une loi normale $\mathcal{N}(\mu_i^k,\ \sigma_i^k)$. Plus explicitement, la variable aléatoire continue Y_t^k suit la densité de probabilité conditionnelle suivante :

$$p_{\,Y_t^k \,|\, X_t^k \,=\, i}(y) = \frac{1}{\sqrt{2\pi}\ \sigma_i^k}\ e^{\,-\frac{(y-\mu_i^k)^2}{{\sigma_i^k}^2}}$$

Comme l'on suppose toutes les réalisations des Y_t^k conditionnellement aux X_t^k indépendantes, il

vient :

$$p_{(Y_t^k)\,|\,(X_t^k)}((y_t^k)) = \prod_{t,k} \frac{1}{\sqrt{2\pi}\,\sigma^k_{x_t^k}}\, e^{-\frac{(y-\mu^k_{x_t^k})^2}{(\sigma^k_{x_t^k})^2}} \tag{44}$$

$$= (\prod_{t,k} \frac{1}{\sqrt{2\pi}\,\sigma^k_{x_t^k}})\, e^{-\sum_{t,k} \frac{(y-\mu^k_{x_t^k})^2}{(\sigma^k_{x_t^k})^2}} \tag{45}$$

$$= e^{-\sum_{t,k}(\log(\sqrt{2\,\pi})+\log(\sigma^k_{x_t^k}))-\sum_{t,k} \frac{(y-\mu^k_{x_t^k})^2}{(\sigma^k_{x_t^k})^2}} \tag{46}$$

$$= \boxed{e^{-\mathcal{U}((y_t^k)\,|\,X)}} \text{ avec } \mathcal{U}((y_t^k)|X) = \sum_{t,k}(\log(\sqrt{2\,\pi}) + \log(\sigma^k_{x_t^k}) + \frac{(y-\mu^k_{x_t^k})^2}{(\sigma^k_{x_t^k})^2}) \tag{47}$$

Loi du processus caché a posteriori Nous allons ici nous intéresser à l'expression de $P((X_t^k)|(Y_t^k))$, car elle nous sera utile, notamment lorsqu'il sera question de trouver la *segmentation* la plus vraissemenblable a posteriori, c'est-à-dire quand on connait déjà l'observation (Y_t^k). Les calculs donnent :

$$P((X_t^k)\,|\,(Y_t^k)) \;=\; P((Y_t^k)|(X_t^k))\frac{P((X_t^k))}{P(Y_t^k)} \tag{48}$$

$$= \boxed{\frac{e^{-\mathcal{U}((y_t^k)\,|\,X) - \underline{\Psi}_{\underline{A}} \cdot \underline{A} - \underline{\Psi}_{\underline{F}} \cdot \underline{F}}}{Z^{prior,\,m}_{\underline{A}\,,\,\underline{F}}\; P(Y_t^k)}} \tag{49}$$

Conclusion partielle Les champs de Markov nous ont fourni un cadre théorique pour modéliser la production d'un signal de parole via un processus caché. A la différence des modèles classiques, notre modèle fait intervenir dans l'énergie de notre champ un terme de synchronisation qui couple sélectivement l'évolution du porcessus caché dans les différentes bandes. Nous verrons dans les prochaines parties comment ce modèle peut être mis en œuvre et nous présenterons ses performances en reconnaissance vocale.

4 L'apprentissage des paramètres du modèle

Dans la partie précédente, nous avons précisé le modèle de parole retenu pour cette étude. Cependant, nous n'avons pas indiqué comment pratiquement ce modèle peut être exploité. En effet, il est bel et bon de prétendre décrire un mot par la donnée des deux familles de coefficients suivantes :

- les paramètres dits *d'attache aux données*, notés $\underline{\lambda}$, égaux aux $(\sigma_i^k, \mu_i^k)_{i,k}$
- les paramètres du modèle caché, notés $\underline{\theta}$: $(\underline{A}\ ,\ \underline{F})$

mais comment obtient-on ces coefficients ? L'expérience ne fournit que des réalisations de chacun des mots, et il n'y a pas d'accès direct possible au modèle caché, qui de toute façon, n'est qu'un modèle possible parmi un infinité d'autres envisageables.
Nous nous attacherons donc dans cette partie à décrire les étapes de la phase préliminaire indispensable à l'étude des signaux de paroles que nous envisageons : la *phase d'apprentissage*.
On pourra retrouver une démarche comparable dans [3].

4.1 Résolution par maximum de vraissemblance

Quelles données pour notre problème ? Comment trouver des valeurs acceptables pour les paramètres $\underline{\theta}$ et $\underline{\lambda}$ d'un mot donné w ? Le premier élément de réponse consiste à se rappeller une des caractéristiques principales du signal de parole : sa grande variabilité. Il faut donc disposer d'un nombre assez important de réalisations de ce mot pour pouvoir prétendre bien le connaître. Nous noterons $(y_t^{k,m})_{m\in 1\ldots M}$ ces réalisations, M étant le nombre de réalisations dont on dispose. Pratiquement, nous avons travaillé avec $M \simeq 50$, une valeur assez courante dans le domaine.
Une fois notre bibliothèque de réalisations constituée, il faut définir une stratégie d'estimation de nos paramètres. Quoi que nous fassions, il ne sera pas possible d'obtenir leur valeur exacte (qui de toute façon n'existe pas, puisqu'il ne s'agit que d'un modèle), et il faut donc faire un choix pour obtenir les valeurs décrivant le "mieux possible" w. Le critère que nous avons retenu est à nouveau celui du *maximum de vraissemblance* : nous allons chercher à résoudre le problème suivant :

$$\text{trouver } (\underline{\theta}\ ,\ \underline{\lambda}) \text{ tel que } (\underline{\theta}\ ,\ \underline{\lambda}) = arg \max_{(\underline{\theta}\ ,\ \underline{\lambda})} P((Y_t^{k,m})_{m\in 1\ldots M} = (y_t^{k,m})_{m\in 1\ldots M} \mid (\underline{\Theta}\ ,\ \underline{\Lambda}) = (\underline{\theta}\ ,\ \underline{\lambda}))$$

cette optimisation étant faite sous les contraintes : $\sigma_i^k \in \mathbb{R}_+^*$, $\mu_i^k \in \mathbb{R}$, $A_i^k \in \mathbb{R}$ et enfin $F_i^{k,l} \in \mathbb{R}$ également.

Nous supposerons les réalisations indépendantes, ce qui est assez réaliste. On obtient alors :

$$P((Y_t^{k,m})_m = (y_t^{k,m})_m \mid (\underline{\Theta}\ ,\ \underline{\Lambda})) = \prod_{m=1}^{M} P((Y_t^{k,m}) = (y_t^{k,m}) \mid (\underline{\Theta}\ ,\ \underline{\Lambda})) \tag{50}$$

$$= \prod_m \sum_{(x_t^k)} P((Y_t^{k,m}) = (y_t^{k,m}) \mid (X_t^k) = (x_t^k)\ ,\ (\underline{\Theta}\ ,\ \underline{\Lambda}))\ P((X_t^k) = (x_t^k) \mid (\underline{\Theta}\ ,\ \underline{\Lambda})) \tag{51}$$

$$= \prod_m \sum_{(x_t^k)} \frac{e^{-\mathcal{U}((y_t^k) \mid X) - \underline{\Psi}_{\underline{A}} \cdot \underline{A} - \underline{\Psi}_{\underline{F}} \cdot \underline{F}}}{Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m}} \tag{52}$$

$$= \prod_m \frac{Z_{\underline{A}\ ,\ \underline{F}}^{post,\ m}}{Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m}} = \frac{\prod_m Z_{\underline{A}\ ,\ \underline{F}}^{post,\ m}}{\prod_m Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m}} \tag{53}$$

où l'on a noté $Z_{\underline{A}\ ,\ \underline{F}}^{post,\ m}$ la fonction de partition associée à notre modèle caché, conditonnellement à la connaissance de la réalisation m du mot w. On voit donc que le problème a une formulation en termes de fonctions de partitions :

$$\boxed{\text{il faut trouver } (\underline{\theta}\ ,\ \underline{\lambda}) \text{ tel que } (\underline{\theta}\ ,\ \underline{\lambda}) = arg\ \underset{(\underline{\theta}\ ,\ \underline{\lambda})}{max}\ \frac{\prod_m Z_{\underline{A}\ ,\ \underline{F}}^{post,\ m}}{\prod_m Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m}}}$$

Cette formulation nous conduit à nous interroger sur le comportement de $Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m}$ en $Z_{\underline{A}\ ,\ \underline{F}}^{post,\ m}$ en fonction de $\underline{A}$ et $\underline{F}$. La proposition suivante nous apporte des informations crutiales à ce sujet.

Proposition 1 *On a les résultats suivants (E et S sont supposés finis) :*

1. $\boxed{\frac{\partial}{\partial\ \underline{A}}\ \log(Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m})\ =\ -\ E_{\ prior,\ m}(\underline{\Psi}_{\ \underline{A}})}$
2. *de même :* $\frac{\partial}{\partial\ \underline{F}}\ \log(Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m})\ =\ -\ E_{\ prior,\ m}(\underline{\Psi}_{\ \underline{F}})$
3. *les mêmes formules sont vraies pour* $Z_{\underline{A}\ ,\ \underline{F}}^{post,\ m}$*, en remplaçant* $E_{\ prior,\ m}$ *par* $E_{\ post,\ m}$
4. $\boxed{\frac{\partial^2}{\partial^2\ \underline{A}}\ \log(Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m})\ =\ \underline{\underline{Cov}}_{\ prior,\ m}(\underline{\Psi}_{\ \underline{A}})}$ *où* $\underline{\underline{Cov}}_{\ prior,\ m}(\underline{\Psi}_{\ \underline{A}})$ *est la matrice des covariances de* $\underline{\Psi}_{\ \underline{A}}$ *pour la loi a priori*
5. *de même :* $\frac{\partial^2}{\partial^2\ \underline{F}}\ \log(Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m})\ =\ \underline{\underline{Cov}}_{\ prior,\ m}(\underline{\Psi}_{\ \underline{F}})$ *où* $\underline{\underline{Cov}}_{\ prior,\ m}(\underline{\Psi}_{\ \underline{F}})$ *est la matrice des covariances de* $\underline{\Psi}_{\ \underline{F}}$ *pour la loi a priori*
6. *à nouveau, on peut transposer ces formules pour les modèles a posteriori.*

Ces résultats sont essentiellement dûs à la dépendance linéaire par rapport aux paramètres des énergies de notre modèèle.

Démonstration Commençons par remarquer que l'opération de dérivation sur la fonction $(\underline{\theta}, \underline{\lambda}) \mapsto \log(Z_{\underline{A}\ ,\ \underline{F}}^{prior,\ m})$ est bien légitime, puisque cette fonction est $\mathcal{C}^\infty$ par rapport à ses variables, sur l'ensemble de son domaine de définition. Pour obtenir le résultat, le principe du raisonnement est simple, comme nous allons le voir. On a :

$\frac{\partial}{\partial \underline{A}} Z_{\underline{A},\underline{F}}^{prior,\,m} = \frac{\partial}{\partial \underline{A}} \sum_{x_t^k} e^{-\underline{\Psi}_{\underline{A}} \cdot \underline{A} - \underline{\Psi}_{\underline{F}} \cdot \underline{F}} = \sum_{x_t^k} -\underline{\Psi}_{\underline{A}} e^{-\underline{\Psi}_{\underline{A}} \cdot \underline{A} - \underline{\Psi}_{\underline{F}} \cdot \underline{F}}$. Donc $\frac{\partial}{\partial \underline{A}} \log(Z_{\underline{A},\underline{F}}^{prior,\,m}) = \frac{\frac{\partial}{\partial \underline{A}} Z_{\underline{A},\underline{F}}^{prior,\,m}}{Z_{\underline{A},\underline{F}}^{prior,\,m}} = -\sum_{x_t^k} \underline{\Psi}_{\underline{A}} P((X_t^k) = (x_t^k)) = -\mathrm{E}_{prior,\,m}(\underline{\Psi}_{\underline{A}})$.
Les autres résultats s'obtiennent en procédant de même.

Munis de ces résultats, nous allons pouvoir énoncer la propostion suivante, qui traite du problème des *points singuliers* de $\frac{\prod_m Z_{\underline{A},\underline{F}}^{post,\,m}}{\prod_m Z_{\underline{A},\underline{F}}^{prior,\,m}}$.

Proposition 2 *Sous les mêmes hypothèses,* $(\underline{\theta}\,,\,\underline{\lambda})$ *est un point singulier de* $\frac{\prod_m Z_{\underline{A},\underline{F}}^{post,\,m}}{\prod_m Z_{\underline{A},\underline{F}}^{prior,\,m}}$ *si et seulement si il vérifie le système d'équations suivant :*

$$\boxed{\begin{cases} \frac{1}{M} \sum_m \boldsymbol{E}_{prior,\,m}(\underline{\Psi}_{\underline{A}}) = \frac{1}{M} \sum_m \boldsymbol{E}_{post,\,m}(\underline{\Psi}_{\underline{A}}) \\ \frac{1}{M} \sum_m \boldsymbol{E}_{prior,\,m}(\underline{\Psi}_{\underline{F}}) = \frac{1}{M} \sum_m \boldsymbol{E}_{post,\,m}(\underline{\Psi}_{\underline{F}}) \\ \forall (i,k) : \ \mu_i^k = \frac{\sum_m \sum_t y_t^{k,m} P_{post,\,m}(X_t^k = i)}{\sum_m \sum_t P_{post,\,m}(X_t^k = i)} \\ \forall (i,k) : \ \sigma_i^k = \frac{\sum_m \sum_t (y_t^{k,m} - \mu_i^k)^2 P_{post,\,m}(X_t^k = i)}{\sum_m \sum_t P_{post,\,m}(X_t^k = i)} \end{cases}}$$

Démonstration Comme notre fonction ne s'annule pas (elle est stictement positive), ses points singuliers sont exactement ceux de son logarithme. La proposition précédente donne immédiatement les 2 premières conditions, par dérivation par rapport à $\underline{\theta}$. Les 2 dernières résultent de la dérivation par rapport à $\underline{\lambda}$. Montrons-le dans le cas de μ_i^k : à un point singulier, on a :

$$0 = \frac{\partial}{\partial\, \mu_i^k} \log(\frac{\prod_m Z_{\underline{A},\underline{F}}^{post,\,m}}{\prod_m Z_{\underline{A},\underline{F}}^{prior,\,m}}) = \sum_m \frac{\partial}{\partial\, \mu_i^k} \log \sum_{t,q} \sum_{(x_t^q)} e^{-\frac{(y_t^{q,m} - \mu_{x_t^q}{}^q)^2}{(\sigma_{x_t^q})^2}} e^{-\underline{\Psi}_{\underline{A}} \cdot \underline{A} - \underline{\Psi}_{\underline{F}} \cdot \underline{F}}$$

$$= -\sum_m \sum_{t,q} \sum_{(x_t^q)} \frac{1}{(\sigma_{x_t^q})^2} \{2(\mu_i^k - y_t^{q,m})\}\, 1_{x_t^q = i \text{ et } q = k}\, e^{-\frac{(y_t^{q,m} - \mu_{x_t^q}{}^q)^2}{(\sigma_{x_t^q})^2}} \frac{e^{-\underline{\Psi}_{\underline{A}} \cdot \underline{A} - \underline{\Psi}_{\underline{F}} \cdot \underline{F}}}{Z_{\underline{A},\underline{F}}^{post,\,m}}.$$

D'où $\mu_i^k \{\sum_m \sum_t P_{post,\,m}(X_t^k = i)\} = \{\sum_m \sum_t y_t^k \ P_{post,\,m}(X_t^k = i)\}$, ce qui fournit le résultat. Un calcul très semblable permet d'obtenir la dernière équation.

Interprétation du système d'équations Les équations précédentes ont une interprétation intéressante. Voici ce qu'on peut en dire :

- les deux premières equations (dites *stochastiques*) signifient qu'à tout point singulier, l'espérance à priori de nos 2 potentiels doivent être égales à leurs espérances a posteriori, id est connaissant les M réalisations de notre mot.
- les moyennes et les variances "optimales" sont données par leurs valeurs "empiriquement pondérées par les modèles a posteriori des réalistions du mot. Elles sont donc définies de façon "auto-cohérente", au sens où elles apparaissent dans les 2 membres de l'équation qui doit les définir.

Commentaire Le problème d'optimisation associé à l'estimation des paramètres par maximum de vraissemblance est difficile. En effet, si la fonction à maximiser est très régulière, il n'en reste pas moins que la recherche d'un suprémum se fait sur un domaine non-compact, sans que l'on puisse facilement s'y ramener dans le cas général. Il n'est donc pas sûr que l'extréma recherché corresponde à un point singulier de la fonction. Quand bien même se serait le cas, la résolution explicite d'un tel système auto-cohérent, hautement non-linéaire, nous semble inenvisageable. Le paragraphe suivant va présenter une méthode classique de résolution approchée du problème, qui nous permettra de contourner cette difficulté.

4.2 Algorithme EM

Démarche Commençons par énoncer le principe EM, baptisé d'après les termes anglais *Expectation-Maximisation*. Il s'agit déterminer les paramètres $(\underline{\lambda}, \underline{\theta})$ vérifiant :

$$(\underline{\lambda}, \underline{\theta}) = arg \max_{(\underline{\lambda},\underline{\theta})} \mathrm{E}_{(\underline{\lambda},\underline{\theta})}(\log(\prod_m P((X_t^k) = (x_t^k), ; \ (Y_t^{k,m}) = (y_t^{k,m}) \mid (\underline{\lambda}, \underline{\theta})))) \tag{54}$$

$$= arg \max_{(\underline{\lambda},\underline{\theta})} \mathrm{E}_{(\underline{\lambda},\underline{\theta})}(\sum_m \log(P((Y_t^{k,m}) = (y_t^{k,m}) \mid (X_t^k) = (x_t^k), ; \ , (\underline{\lambda}, \underline{\theta}))) + \log((X_t^k) = (x_t^k), ; \ , \mid (\underline{\lambda}, \underline{\theta}))) \tag{55}$$

Ainsi, la maximisation se fait sur l'espèrance du logarithme de la probabilité de la probabilité conjointe des deux processus, les observations $(y_t^{k,m})$ étant connues. Ce problème, à l'instar du problème précédent, est fort difficile à résoudre. C'est pourquoi on le résoud de la manière itérative. Pour cela, on définit :

$$Q(\underline{\theta}, \underline{\lambda}, \underline{\theta}_n, \underline{\lambda}_n) = \mathrm{E}_{\underline{\theta}_n,\underline{\lambda}_n}(\sum_n \log(P((X_t^k) = (x_t^k), ; \ (Y_t^{k,m}) = (y_t^{k,m}) \mid (\underline{\lambda}, \underline{\theta}))))$$

Q étant une fonction de "qualité". Le problème EM s'écrit :

$$\boxed{(\underline{\lambda}, \underline{\theta}) = arg \max_{(\underline{\lambda},\underline{\theta})} Q(\underline{\theta}, \underline{\lambda}, \underline{\theta}, \underline{\lambda})}$$

et pour contourner la difficulté de maximiser à la fois sous l'espérance et dans la probabilité, nous allons "séparer" ces variables et trouver une suite de paramètres $(\underline{\lambda}_n, \underline{\theta}_n)_n$ définie par récurrence :

$$(\underline{\lambda}_{n+1}, \underline{\theta}_{n+1}) = arg \max_{(\underline{\lambda},\underline{\theta})} Q(\underline{\theta}, \underline{\lambda}, \underline{\theta}_n, \underline{\lambda}_n)$$

Cette optimisation donne lieu à deux maximisations indépendantes l'une de l'autres, puisque $\log(P((X_t^k) = (x_t^k), ; \ (Y_t^{k,m}) = (y_t^{k,m}) \mid (\underline{\lambda}, \underline{\theta})))$ se sépare en deux termes grâce à la formule de Bayes. Pour passer du rang n au rang $n+1$, on résoud donc :

$$\begin{cases} \underline{\theta}_{n+1} = arg \max_{\underline{\theta}} \ \log(P((X_t^k) = (x_t^k) \mid \underline{\theta}_n)) \\ \underline{\lambda}_{n+1} = arg \max_{\underline{\lambda}} \ \log(P((Y_t^k) = (y_t^k) \mid (X_t^k) = (x_t^k), \underline{\lambda}_n)) \end{cases}$$

En vertu des résultats de dérivation obtenus au paragraphe précédent, le système se réécrit (en admettant qu'il se ramène à la recherche d'un point singulier) :

$$\left\{\begin{array}{rcl} \frac{1}{M}\sum_m \ \mathrm{E}_{\underline{\theta}_{n+1}}(\underline{\Psi}_{\ \underline{A}}) & = & \frac{1}{M}\sum_m \ \mathrm{E}_{(\underline{\lambda}_n,\ \underline{\theta}_n),\ m}(\underline{\Psi}_{\ \underline{A}}) \\ \frac{1}{M}\sum_m \ \mathrm{E}_{\underline{\theta}_{n+1}}(\underline{\Psi}_{\ \underline{F}}) & = & \frac{1}{M}\sum_m \ \mathrm{E}_{(\underline{\lambda}_n,\ \underline{\theta}_n),\ m}(\underline{\Psi}_{\ \underline{F}}) \\ \forall (i,k):\ \mu_i^k & = & \frac{\sum_m \sum_t y_t^{k,m}\, P_{(\underline{\lambda}_n,\ \underline{\theta}_n),\ m}(X_t^k=i)}{\sum_m \sum_t P_{(\underline{\lambda}_n,\ \underline{\theta}_n),\ m}(X_t^k=i)} \\ \forall (i,k):\ \sigma_i^k & = & \frac{\sum_m \sum_t (y_t^{k,m}-\mu_i^k)^2\, P_{(\underline{\lambda}_n,\ \underline{\theta}_n),\ m}(X_t^k=i)}{\sum_m \sum_t P_{(\underline{\lambda}_n,\ \underline{\theta}_n),\ m}(X_t^k=i)} \end{array}\right.$$

D'après [3], ce procédé itératif permet d'aboutir à un maximum local de la fonction Q. Si l'on est resté dans un espace compact de paramètres, le passage à $n \longrightarrow +\infty$ dans le système précédant nous assure que nous sommes à un point singulier de la fonction devant être maximisée pour aboutir au maximum de vraissemblance : on a aboutit à un maximum local de celle-ci.

Résolution numérique Comment résoudre le problème précédent ? Est-il bien posé ? Nous allons aborder ces questions dans ce paragraphe, sans prétendre y apporter une réponse entièrement rigoureuse. Voici ce que l'on peut on dire :

- pour les paramètres d'attache aux données, le problème est bien posé, puisque les moyennes et les variances sont donnés par une formule !
- le problème est un peu plus complexe pour les paramètres de notre modèle caché. Il faut en effet résoudre deux équations *stochastiques*, c'est-à-dire de la forme :$\mathrm{E}_{THE_{n+1}}(\underline{\Psi}) = \underline{X}$. Nous admettrons qu'elle définit une unique solution, ce qui doit découler la propriété de définition positive de son gradient (qui est une matrice de covariance). On a ici posé $\underline{\Psi} = \sum_m (\underline{\Psi}_{\ \underline{A}}{}^m,\ \underline{\Psi}_{\ \underline{F}}{}^m)$.

En pratique, le problème est résolu comme suit :

- estimation des moyennes et des variances par simulation du processus caché pour les coefficients $(\underline{\lambda}_n,\ \underline{\theta}_n)$
- résolution approchée de l'équation stochastique par un algorithme de gradient

Le gradient stochastique Une démarche classique pour résoudre une équation du type $f(x) = y$ consiste à lui appliquer un algorthme de Newton, qui donne une suite de solutions approchées (x_n) comme suit :

$$x_{n+1} = x_n + \frac{(y - f(x_n))}{f'(x_n)}$$

La généralisation de cet algorithme au cas multidimensionnel est donnée par :

$$\underline{\theta}_{n+1} = \underline{\theta}_n + (\frac{\partial}{\partial \underline{\theta}}\ \mathrm{E}_{\underline{\theta}_n}(\underline{\Psi}))^{-1}(\underline{X} - E_{\underline{\theta}_n}(\underline{\Psi})) \tag{56}$$

$$= \underline{\theta}_n - (\underline{\underline{\mathrm{Cov}}}\ (\underline{\Psi}))^{-1}(\underline{X} - E_{\underline{\theta}_n}(\underline{\Psi})) \tag{57}$$

Ce problème est là encore assez délicat à résoudre en pratique, puisque la dimension de cette matrice est élevée, puisque pour $\underline{\Psi}_{\ \underline{F}}$, c'est une matrice carrée de $24 * 24 * 14 = 8064$ de côté ! On peut effectuer une approximation supplémentaire en pratique : à savoir négliger les corrélations

entre les différents potentiels pour ne garder que les termes diagonaux. on parle *d'approximation diagonale*. Voici ce qu'elle donne :

$$\begin{cases} \forall i,k:\ (A_i^k)_{n+1} = (A_i^k)_n + \frac{1}{n^\gamma} \frac{\sum_m \mathrm{E}_{\underline{\theta}_n,\ m}(D_i^k) - \mathrm{E}_{(\underline{\theta}_n,\underline{\lambda}_n),\ m}(D_i^k)}{\sum_m Var_{\underline{\theta}_n,\ m}(D_i^k)} \\ \\ \forall i,k,l:\ (F_i^{k,l})_{n+1} = (F_i^{k,l})_n + \frac{1}{n^\gamma} \frac{\sum_m \mathrm{E}_{\underline{\theta}_n,\ m}(|D_i^k - D_i^l|^\beta) - \mathrm{E}_{(\underline{\theta}_n,\underline{\lambda}_n),\ m}(|D_i^k - D_i^l|^\beta)}{\sum_m Var_{\underline{\theta}_n,\ m}(|D_i^k - D_i^l|^\beta)} \end{cases}$$

Dans l'algorithme d'estimation des paramètres implémenté, c'est l'algorithme qui a été retenu. Pour pouvoir le faire fonctionner, il a été nécessaire de simuler parallèlement le processus caché pour chacune des réalisations du mot, et ce pour sa loi a priori et sa loi a posteriori. Les estimateurs de chacune des espérances mentionnées ici ont simplement été pris comme la moyenne algébrique des grandeurs correspondantes dans les échantillons fournis par la simulation.
Enfin, on remarquera qu'un facteur multiplicatif $\frac{1}{n^\gamma}$ avec $\gamma \in [0.5,\ 1]$ est ajouté afin d'accélérer la convergence (confer [3]). En effet, celle-ci n'est pas complètement garantie, en raison de l'approximation diagonale retenue. N'ayant pas eu l'opportunité d'implémenter une autre variante de résolution du gradient stochastique, nous ne discuterons pas plus avant cette méthode de calcul.

Appellation "EM" Pourquoi cet algorithme a-t-il été baptisé ainsi ? La réponse réside dans le déroulement de l'algorithme. En effet, on a vu que la résolution des équations précédentes demandaient 2 étapes : une dite d'"estimation" (expectation) où l'on estime selon la loi à l'étape n un certain nombre de grandeurs, et une autre, de "maximisation" où l'on en déduit les paramètres à l'étape $n+1$, soit par un gradient stochastique, soit par le calcul d'une moyenne.

4.3 Une initialisation possible pour un potentiel de synchronisation quadratique

Nous venons de voir que l'algorithme EM suppose une initialisation des paramètres donnée, $(\underline{\lambda}_0,\ \underline{\theta}_0)$. De cette initialisation dépendra la valeur finale des paramètres obtenus par l'algorithme, car on n'obtient par cette appproche qu'un maximum local de notre vraissemblance. Comment effectuer l'initialisation de nos paramètres ? Commençons par présenter une initialisation très simple :

1. sur chaque bande de fréquence, on obtient des moyennes et des variances empiriques en subdivisant uniformément l'intervalle $1 \ldots T$. Ceci se justifie par le fait que les différents sons ont des longueurs de grandeur comparables.
2. on pose $\underline{\lambda}_0 = 0$: cela revient à supposer que leur importance est faible : les différents sons sont de longueur comparable, et les bandes peu couplées.

Nous donnerons dans la partie suivante les performances de cette initialisation en reconnaissance vocale, c'est-à-dire si on n'effectue pas une estimation plus précise. Présentons maintenant une autre initialisation, proposée par Marc Sigelle. En voici le principe :

1. on estime empiriquement les moyennes et les variances comme précédemment
2. puis on recherche, sur chacun des signaux, la segmentation la plus vraissemblable pour ces paramètres, par un recuit simulé

3. on estime à nouveau empiriquement les moyennes et les variances, cette fois sur les nouvelles segmentations
4. enfin, on estime $\underline{\lambda}$ en effectuant la première étape d'une estimation par gradient stochastique. Pour $\beta = 2$, ce calcul est particulièrement précis.

En effet, en négligeant la corrélation entre les différents coefficients, l'on obtient (comme cela a été montré au paragraphe précédent) :

$$\forall i,k: \ \underline{A}_i^k \ = 0 \ + \ \frac{\sum_m \mathrm{E}_0 \left((\underline{\Psi}_{\underline{A}}) \right)_i^k) - (\underline{\Psi}_{\underline{A}})_i^k}{\sum_m \mathrm{Var}((\underline{\Psi}_{\underline{A}})_i^k)}$$

et

$$\forall i,k,l: \ \underline{F}_i^{k,l} \ = 0 \ + \ \frac{\sum_m \mathrm{E}_0(\ (\underline{\Psi}_{\underline{F}})_i^{k,l}) - (\underline{\Psi}_{\underline{F}})_i^{k,l}}{\sum_m \mathrm{Var}((\underline{\Psi}_{\underline{F}})_i^{k,l})}$$

Or, pour $\beta = 2$, $\underline{\Psi}_{\underline{A}} = (D_i^k)$ et $\underline{\Psi}_{\underline{F}} = ((D_i^k - D_i^l)^2)$. Et nous avons vu dans la partie 2 que nous pouvions les calculer. C'est pourquoi dans ce cas précis, il est très intéressant d'utiliser cette initialisation. Nous verrons dans la partie suivante les résultats obtenus à ce sujet.

5 La phase de reconnaissance

Dans cette partie, nous détaillons les calculs qui doivent être menés à bien lors de la phase de reconnaissance. Est présenté également un calcul original de fonction de partition, développé durant le stage, ainsi que son application à la reconnaissance pratique. Enfin, est brièvement décrite la méthodologie adoptée pour l'analyse des (très !) nombreuses expérimentations numériques menées durant le stage.

Le cadre théorique présenté dans les parties précédentes a permis la mise au point d'un logiciel de reconnaissance vocale utilisé avec les restrictions suivantes dans cette étude :

- les mots doivent être isolés
- les réalisations provenaient d'un unique locuteur. Cette rstriction n'avait pas de caractère obligatoire, et le modèle peut très utilisé en mode multi-locuteur. Mais encore faudra-t-il alors avoir effectué un apprentissage des paramètres adéquat...

Ce logiciel a été programmé à partir d'une première version développée par Marc Sigelle, comprenant les structures de données principales, ainsi que l'algorithme du recuit simulé. Pour être pleinement opérationnel, le logiciel a été adapté aux besoins des techniques présentées dans ce rapport (calcul des fonctions de partitions, algorithme EM, gradient stochastique..) et un effort particulier a été fourni concernant la présentation et l'analyse des résultats obtenus, comme on peut le voir dans l'annexe correspondante.

5.1 Déroulement de la phase de reconnaissance

Avant de présenter les résultats obtenus et les conclusions qui peuvent en être tirées, terminons la traitement théorique du problème en analysant le déroulement de la phase de reconnaissance proprement dite. Comme on l'a montré dans la première partie, on reconnaitra le mot w à la lecture du signal (Y_t^k) si et seulement si :

$$w = arg \max_w LS(w)$$

Dans le cadre de notre modèle markovien, avec l'approximation de Viterbi, cela donne :

$$w = arg \max_w (\prod_{t,k} P(Y_t^k = y_t^k \mid X_t^k = x_w^{*k}{}_t,\ W = w))\ )\ P((X_t^k) = x_w^* \mid W = w) \tag{58}$$

$$= arg \max_w e^{-\mathcal{U}((Y_t^k) \mid x_w^*)} \frac{e^{-U_{HMM}(x_w^* \mid W=w) - U_{synchro}(x_w^* \mid W=w)}}{Z_{\underline{A}\,,\,\underline{F}}} \tag{59}$$

Connaissant les paramètres $(\underline{\theta},\ \underline{\lambda})$ grâce à l'estimation des paramètres réalisée via un algorithme EM, il est très simple d'estimer tous les termes de l'expression sauf... la fonction de partition a priori ! Il est nécessaire à ce stade de se livrer à nouveau à des approximations.

Un développement limité Notre objectif est d'aboutir à une approximation de $Z_{\underline{A}\ ,\ \underline{F}}$. Pour cela, nous allons nous baser sur la remarque suivante :

$$Z_{\underline{A}\ ,\ \underline{F}} \ = \ \sum_{(x_t^k)} \sum_{t,k} \ e^{-U_{HMM}(x_w^* \mid W=w) - U_{synchro}(x_w^* \mid W=w)} \tag{60}$$

$$= \ Z_{\underline{A}} \sum_{(x_t^k)} \sum_{t,k} \ \frac{e^{-U_{HMM}(x_w^* \mid W=w)}}{Z_{\underline{A}}} \ e^{-U_{synchro}(x_w^* \mid W=w)} \tag{61}$$

$$= \boxed{Z_{\underline{A}} \ \mathrm{E}_{\ \underline{A}} \ (e^{-U_{synchro}(x_w^* \mid W=w)})} \tag{62}$$

Or, les coefficients de synchro sont relativement petits (de l'ordre de 10^{-3} ou 10^{-4}). Pour effectuer une approximation, nous allons nous baser sur une technique de développement limité autour de 1 par rapport à $\underline{F}$ de notre fonction de $\underline{F}$ définie par une espérance. On trouve classiquement :

$$Z_{\underline{A}\ ,\ \underline{F}} \ \simeq \ Z_{\underline{A}} \ e^{-\underline{F}.\mathrm{E}_{\ \underline{A}}(\underline{\Psi}_{\ \underline{F}})}$$

où $\mathrm{E}_{\ \underline{A}}(\underline{\Psi}_{\ \underline{F}})$ est l'espérance de $\underline{\Psi}_{\ \underline{F}}$ pour la loi du champ de Gibbs où l'on ne prend en compte que l'énergie couplage intra-bandes. Comme les coefficients $\underline{A}$ sont également assez petits (1×10^{-2} au plus), on prend en partique pour $\beta = 2$: $\mathrm{E}_{\ \underline{A}}(\underline{\Psi}_{\ \underline{F}}) \simeq \mathrm{E}_0 \ (\underline{\Psi}_{\ \underline{F}})$, où l'on note 0 la loi de probabilité uniforme sur notre modèle caché contraint. Car on sait en effet calculer cette espérance en vertu des résultats de la partie 2. Dans l'autre cas, nous utilisons des simulations pour estimer cette grandeur.

Au final, nos approximations donnent :

$$LS \ \simeq \ e^{-\mathcal{U}((Y_t^k) \mid x_w^*)} \frac{e^{-U_{HMM}(x_w^* \mid W=w) - U_{synchro}(x_w^* \mid W=w)}}{Z_{\underline{A}} \ e^{-\underline{F}.\mathrm{E}_{\underline{A}}\ (\underline{\Psi}_{\ \underline{F}})}}$$

en général, et pour $\beta = 2$ peut faire une approximation supplémentaire utilisant les résultats de la partie 2 :

$$LS \ \simeq \ e^{-\mathcal{U}((Y_t^k) \mid x_w^*)} \frac{e^{-U_{HMM}(x_w^* \mid W=w) - U_{synchro}(x_w^* \mid W=w)}}{Z_{\underline{A}} \ e^{-\underline{F}.\mathrm{E}_0\ (\underline{\Psi}_{\ \underline{F}})}}$$

Deux possibilités s'offrent alors à nous pour effectuer informatiquement le calcul :

1. soit l'on sait calculer $Z_{\underline{A}}$ et on l'utilise son expression exacte
2. soit on utilise à nouveau un développement par cumulants pour estimer $Z_{\underline{A}}$

Plus précisément, le développement par cumulants de $Z_{\underline{A}}$ donne :

$$Z_{\underline{A}} = Z_0 exp - \underline{A}.\mathrm{E}_0(\ \underline{\Psi}_{\ \underline{A}}) \text{ à l'ordre 1}$$

5.2 Un calcul exact de fonction de partition

Une méthode calcul originale a été développée pour calculer $Z_{\underline{A}}$ dans un cas particulier, mais assez général : celui où les p_i sont distincts 2 à 2. Rappelons que la matrice de transition $\mathbb{P}_k$ de chacune des K chaînes de Markov indépendantes se présente comme suit :

$$\mathbb{P}_k = \begin{pmatrix} p_1^k & 1-p_1^k & 0 & 0 & \dots & 0 \\ 0 & p_2^k & 1-p_2^k & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & & \vdots \\ \vdots & & & & & \\ \vdots & & & \ddots & p_N^k & 1-p_N^k \\ 0 & \dots & & & 0 & 1 \end{pmatrix}$$

Si notre hypothèse est vérifiée pour la bande k, la matrice $\mathbb{P}_k$ est diagonalisable, puisqu'elle a $N+1$ valeurs propres distinctes. Elle s'écrit sous forme factorisée en :

$$\mathbb{P}_k = \mathbb{Q}_k \mathbb{D}_k (\mathbb{Q}_k)^{-1}$$

où $(\mathbb{Q}_k)$ est la matrice des vecteurs propres de $\mathbb{P}_k$ dans la base canonique de $\mathbb{R}^{N+1}$. Cette propriété de diagonalisation est très utile, puisqu'elle permet de calculer simplement la fonction de partition ${Z_{\underline{A}}}^k$. En effet, on a la propriété fondamentale :

$$\boxed{{Z_{\underline{A}}}^k = P(X_T^k = N | X_1^k = 1) = \mathbb{P}_k^{T-1}(1, N)} \text{ (confer partie 2)}$$

Pour calculer ${Z_{\underline{A}}}^k$, on voit alors qu'il suffit de connaitre l'expression de $\mathbb{P}_k^{T-1}$, ce qui est particulièrement simple dans le cas où $\mathbb{P}_k$ est diagonalisable, puisqu'alors :

$$\mathbb{P}_k^{T-1} = (\mathbb{Q}_k \mathbb{D}_k (\mathbb{Q}_k)^{-1})^{T-1} \tag{63}$$

$$= \mathbb{Q}_k (\mathbb{D}_k)^{T-1} (\mathbb{Q}_k)^{-1} \tag{64}$$

avec

$$(\mathbb{D}_k)^{T-1} = \begin{pmatrix} (p_1^k)^{T-1} & 0 & 0 & 0 & \dots & 0 \\ 0 & (p_2^k)^{T-1} & 0 & 0 & \dots & 0 \\ \vdots & 0 & \ddots & \ddots & & \vdots \\ \vdots & & & & & \\ \vdots & & & \ddots & (p_N^k)^{T-1} & 0 \\ 0 & \dots & & & 0 & 1 \end{pmatrix}$$

Calcul des vecteurs propres Pour plus de lisibilité, dans cette partie, nous omettrons les indices k de bande. Pour calculer $Z_{\underline{A}}$, il nous faut détermniner les vacteurs propres de $\mathbb{P}$. Et donc résoudre pour tout $i \in 1 \dots N$, trouver un $\underline{V}_n \in \mathbb{R}^N$ tel que :

$$\mathbb{P} \times \underline{V}_n \ = \ p_n \underline{V}_n$$

Proposition 1 *Une solution de ce système d'équation est donnée par :*

$$\boxed{\begin{cases} \forall j \in 1 \dots N+1 : \underline{V}_n(j) = 0,\ \forall j > n, \text{ avec la convention } p_{N+1} = 1 \\ \forall j \in 1 \dots N+1 : \underline{V}_n(n) = 1 \\ \forall j \in 1 \dots N+1 : \underline{V}_n(j) = \frac{1-p_j}{p_n - p_j} \underline{V}_n(j+1) \end{cases}}$$

Pour obtenir cette solution, il suffit de poser le système à résoudre.

On obtient donc une matrice $\mathbb{Q}$ triangulaire supérieure, avec uniquement des 1 sur sa diagonale. Se pose enfin le problème de l'inversion de cette matrice... Qu'il n'est pas nécessaire de connaitre entièrement, puisque l'on ne s'intéresse qu'à un seul coefficient de $\mathbb{P}^{T-1}$: il suffit de connaitre son avant-dernière colonne.

Proposition 2 *Soit $\underline{Y} = (q_1, , q_2, \dots, q_{N+1})$ le $N^{ième}$ vecteur colonne de $(\mathbb{Q})^{-1}$. Ses coefficients sont donnés par :*

$$\boxed{\begin{cases} q_{N+1} = 0 \\ q_N = 1 \\ \forall j \in 1 \dots N-1 : q_j = -\sum_{i=j+1}^{N} q_i \, \underline{V}_i(j) \end{cases}}$$

Là encore, il suffit de poser les équations du problème à résoudre.

Au final, les résultats précédents mis bout à bout donnent :

$$\boxed{Z_{\underline{A}} = \sum_{i=1}^{N} \underline{V}_i(1) \; q_i \; (p_i)^{T-1}}$$

Pour obtenir la fonction de partition globale à partir des fonctions de partition pour chaque bande, il ne reste plus qu'à en faire le produit, puisque les bandes sont par hypothèse indépendantes.

Limitations à l'utilisation de cette formule Malheureusement, cette formule présente deux inconvénients majeurs, liés l'un à l'autre. Ils ont sérieusement limité son application pratique pour la reconnaissance vocale. Les voici :

- elle n'est valable que pour des p_i distincts deux à deux dans chaque bande
- a première vu, ce cas de figure est assez improbable, puisque $N \simeq 10$ et que la précision sur les "long double" (avec lesquels le programme a été écrit en C) est très grande. Mais il est fortement lié au module des coefficients des matrices : plus les p_i seront proches, plus les coefficients atteindront des tailles énormes. On est alors confronté numériquement à des problèmes très difficilement surmontables : les erreurs d'arrondi rendent le calcul de $Z_{\underline{A}}$ impossible.

Nous avons tout de même pu calculer $Z_{\underline{A}}$ pour les mots courts (où N est donc petit), et donc pu appliquer les calculs dans le cadre de la reconnaissance vocale. Cependant, tous les mots du dictionnaires ne sont alors pas traités de la même manière, ce qui a priori rend difficile l'interprétation des résultats obtenus par cette méthode "panachée".

5.3 L'analyse des résultats

Implémentation du modèle Une partie très importante du stage proprement dit à consister à programmer dans le langage C un logiciel de reconnaissance vocale utilisant le modèle markovien décrit. De nombreuses expérimentations numériques ont pu avoir lieu, permettant d'apporter une réponse à de nombreux problèmes pratiques. Nous avons choisi volontairement de ne pas présenter la structure de données retenue, car elle n'était qu'un simple outil mis au service de l'étude du modèle original de parole présenté.

Les variantes de l'algorithme et les questions qu'elles soulèvent Comme on a pu le constater à la lecture de ce mémoire, le modèle de parole proposé possède un grand nombre de degrés de liberté, au sens où sa mise en œuvre peut se faire avec de multiples variantes. Avant de présenter les résultats proprement dit, commençons par en dresser l'inventaire, ce qui nous amènera naturellement à formuler un certain nombre de questions :

- **initialisation** de l'algorithme d'apprentissage des paramètres : plusieurs possibilités ont été envisagées, dont l'initialisation présentée en fin de dernière partie. Laquelle choisir ?
- **déroulement de l'EM** : combien d'itérations effectuer dans chaque algorithme utilisé ? Autre formulation : quelle vitesse de convergence pour l'apprentissage des paramètres ?
- **choix de** β : quelle valeur adopter ? Nous avons vu que le cas $\beta = 2$ facilite les calculs. Faut-il adopter cette valeur ?
- **verticale-connexité** : est-il nécessaire de conserver un grand nombre de coefficients de synchronisation, dont le maniement est pénible numériquement ? Si l'on répond par la négative, lesquels retenir ? Nous formulerons une réponse en terme de *connexité verticale*, comme nous le verrons plus tard.
- **calcul de la fonction de partition** : nous avons vu au début de cette partie qu'un calcul moins approximatif pouvait être envisagé. Qu'en est-il en pratique ?

Ces différents points nous donneront un premier point de vue sur la maniabilité et la validité du modèle de parole original que nous étudions ici.

Protocole expérimental L'ensemble des expériences effectuées l'ont été à partir d'une base de données de 10 mots, dont on disposait d'une centaine de réalisations pour chacun. A chaque fois, une cinquantaine de réalisations ont été utilisés pour les besoins de la phase d'apprentissage, et les réalisations restant ont servi de tests au logiciel ainsi étalonné.

Analyse fine des résultats Afin de pouvoir mieux interpréter les résultats des séries de tests effectuées pour chacune des variantes retenues, une présentation des données commode a été mise au point. On s'est appuyé pour cela sur la décomposition de notre score acoustique en plusieurs termes. Ces termes proviennent de la prise du logarithme sur notre score acoustique. Par abus de langage, nous continuerons à parler de "score" alors qu'il s'agit à proprement parler du logarithme des scores :

$$\log(LS) \simeq \log(e^{-\mathcal{U}((Y_t^k) \mid x_w^*,\, T=t)} \frac{e^{-U_{HMM}(x_w^* \mid W=w,\, T=t) - U_{synchro}(x_w^* \mid W=w,\, T=t)}}{Z_{\underline{A}\,,\,\underline{F}}^{prior,\, m}} P(T=t|W=w)$$

$$= \underbrace{-\mathcal{U}((Y_t^k) \mid \underset{w}{x^*},\, T=t)}_{\text{attache aux données}} \underbrace{-U_{HMM}(\underset{w}{x^*} \mid W=w,\, T=t)}_{\text{score intra-bande}} \underbrace{-U_{synchro}(\underset{w}{x^*} \mid W=w,\, T=t)}_{\text{synchronisation}}$$

$$\underbrace{-\log(Z_{\underline{A}\,,\,\underline{F}}^{prior,\,m})}_{\text{fonction de partition}} + \underbrace{log(P(T=t \mid W=w))}_{\text{durée}}$$

Ainsi, notre score s'écrit comme somme de nombreux scores partiels. Dans les expériences qui ont été menées, ont été utilisés des scores qui sont les mêmes que ceux qui viennent d'être présentés, à ceci prêt que l'on a ajouté aux scores "intra-bande" et de "synchronisation" le terme correspondant provennant du développement en cumulants de $Z_{\underline{A}\,,\,\underline{F}}$. Cela s'écrit :

$$\begin{cases} \text{score intra-bande} \;=\; -U_{HMM}(\underset{w}{x^*} \mid W=w,\, T=t) + \mathrm{E}_0(\underline{\Psi}_{\,\underline{A}}).\underline{A} \\ \text{score de synchronisation} \;=\; -U_{synchro}(\underset{w}{x^*} \mid W=w,\, T=t) + \mathrm{E}_{\underline{A}}(\underline{\Psi}_{\,\underline{F}}).\underline{F} \end{cases}$$

Afin de rendre plus exploitables - et comparables - les résultats expérimentaux, on procède pour chaque test à des caculs statistiques qui permettent d'avoir accès aux informations suivantes :

- pour chaque réalisation m d'un mot w, sont calculé le rang de chacun des différents scores pour ce mot, par rapport aux autres mots du dictionnaire
- est calculé un rang global, permettant de savoir si le mot a été reconnu ou pas (si tel est le cas, le rang sera bien sûr 1). On présente aussi les rangs du score partiel de "segmentation" (tous les scores sauf l'attache qux données et la durée).
- pour un mot donné, on présente à la fin des tests les rangs moyens pour chacun des scores, ainsi que le rang moyen du mot.
- un bilan global est réalisé à la fin des test de tous les mots, afin de juger des performances globles du logiciel.

Exemple Pour 5 étapes d'EM, 1 seule étape de gradient stochastique, une connexité verticale de 4 et l'initialisation la plus simple, on obtient par exemple pour le mot "concert" :

```
       Detailled statistics for concert :
rank  1 :  98 per, data:  98 per, segmentation:   0 per, duree:  32, intra:  20 per, synchro  8
rank  2 :   2 per, data:   2 per, segmentation:   0 per, duree:   4, intra:  22 per, synchro  1
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  24, intra:  12 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:  88 per, duree:  20, intra:  20 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:  10 per, duree:  10, intra:  14 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   2 per, duree:   0, intra:  12 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   8, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 4.1, duree: 3.2, intra: 3.2, synchro 1.2.
```

On voit ainsi que le mot a été reconnu dans 98 pourcent des cas, et que parmis tous les scores, c'est celui de l'attache aux données permet une si bonne reconnaissance du signal : il est pratiquement toujours en première position.

6 Les résultats expérimentaux

Pour conclure, une synthèse des principaux résultats et enseignements apportés par le stage. On voit ainsi d'une part quells sont les performances du modèle, sans bruit comme en présence de bruit. D'autre part, plusieurs réponses importantes concernant l'optimisation du modèle sont apportées.

Nous allons tâcher ici d'apporter des élements de réponse aux questions soulevées à la fin de la partie précédente, à partir des résultats obtenus grâce aux expérimentations numériques.

6.1 Les performances globales du modèle

Un taux de reconnaissance élevé Quelles performances peut-on attendre de notre modèle ? Actuellement, les logiciels de reconnaissance vocale basé uniquement sur les HMM obtiennent des résultats excellents dans les conditions où nous nous plaçons, à savoir la reconnaissance de mots isolés, pronnoncés par un unique locuteur, avec une bonne qualité d'enregistrement (un rapport signal sur bruit faible). On obtient ainsi un taux de reconnaissance de l'ordre de 99 à 100 pourcents. Voici les résultats obtenus dans différentes configurations :

NiterEM :	5	10
Cas 1	$94,2\ \%$	$95,0\ \%$
Cas 2	$98,4\ \%$	$98,2\ \%$

On voit ainsi que l'on obtient assez aisément des taux de reconnaissance élevés, très proches de 100. Et dans les 2 cas de figure, la durée de la phase d'apprentissage n'était pas excessive, de l'ordre de 10 minutes sur une station de travail des années $97-98$ (Sun Ultra 5), ceci correspondant à 5 ou 10 itérations pour notre algorithme EM. Ainsi, notre modèle de reconnaissance vocal fonctionne bien.

Contributions des différents scores La reconnaissance s'effectue correctement, mais il est intéressant de se demander quelles sont les contributions au score qui sont prépondérantes. Pour cela, intéressons-nous aux statistiques obtenues pour un mot donné, à savoir "quitter", pour une variante assez peu coûteuse (10 itérations pour l'EM) :

```
       Detailled statistics for quitter :
rank  1 :  76 per, data: 100 per, segmentation:   0 per, duree:   0, intra:  10 per, synchro
rank  2 :  24 per, data:   0 per, segmentation:  62 per, duree:  72, intra:  48 per, synchro  4
rank  3 :   0 per, data:   0 per, segmentation:  38 per, duree:  10, intra:  20 per, synchro  2
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   8 per, synchro  1
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:  14, intra:   4 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   4 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro

On average: rank: 1.2, data: 1.0, segmentation: 2.4, duree: 2.6, intra: 2.9, synchro 3.1.
```

De cet exemple, très caractéristiques des résultats que l'on obtient on général, on peut tirer plusieurs enseignements :

1. le score d'attache aux données est de loin celui qui caractérise le mieux le signal à reconnaître
2. les scores intra-bande et de synchronisation n'offrent pas de meilleurs performances que le score de durée
3. malgré de très bons scores d'attache aux données, la reconnaissance échoue dans 24 pourcents des cas

On voit ainsi que les paramètres d'attache aux données caractérisent bien le mot : ils ont été correctement estimés. Par contre, la segementation du signal est imparfaite : on peut l'interpréter comme dûe à une estimation des paramètres intra-bande et d'attache aux données médiocre. Nous mettons ici le doigt sur un enseignement important de cette étude : *les paramètres du modèle caché sont - de loin - les plus difficiles à estimer.*

6.2 La phase d'initialisation

Les différentes possibilités Voici une liste des variantes qui ont été étudiées durant la durée du stage :

- l'initialisation présentée à la fin de la $4^{\text{ième}}$ partie, que nous noterons $INI = 1$
- une initialisation beaucoup plus simple : les moyennes et les variances sont estimés empiriquement à partir de la segmentation uniforme, et les autres coefficients sont mis à 0. Nous la noterons $INI = 0$
- une initialisation intermédiaire a été étudiée, notée $INI = 0.5$, il s'agit de la première initialisation, à ceci près que les coefficients de synchronisation ne sont pas estimés (et donc nuls), au contraire des coefficients intra-bande.

La question de l'initialisation pour β quelconque étant difficile, nous ne traitons ici d'abord le cas $\beta = 2$, plus simple à traiter informatiquement, puisqu'il fait moins appel aux simulations, comme nous l'avons vu.

Comment juger de l'efficacité d'une initialisation ? Un premier critère consiste à effectuer un test de reconnaissance *directement* à partir des coefficients *issus de l'initialisation.* On obtient, lorsque l'on se place en 4-connexité verticale (cette notion sera définie dans un paragraphe ultérieur) :

INI :	0	0.5	1
Reconnaissance :	$94.2\ \%$	$96.4\ \%$	$74.9\ \%$

On voit ainsi que pour les deux premières initialisations, on obtient immédiatement des performances correctes, meilleures dans le cas où les paramètres intra-bandes interviennent. On peut en conclure que *l'initialisation des paramètres intra-bande est bonne.* Par contre, tel n'est pas le cas pour les paramètres de synchronisation : la reconnaissance est immédiatement moins efficace. Pour obtenir de bonnes performances avec la synchronisation, le prix à payer en calculs est assez important.

L'initialisation la plus performante On peut également juger de l'initialisation en regard des meilleurs résultats obtenus grâce à elle. Pour des raisons qui seront expliquées au paragraphe correspondant, nous avons retenu la 4-connexité verticale, 5 iterations pour chaque gradient stochastique au sein de l'EM, et un nombre élevé d'itérations pour l'EM : 40. Voici ce que l'on obtient :

INI :	0	0.5	1
Reconnaissance :	$91.2\,\%$	$98.8\,\%$	$94.0\,\%$

On voit ainsi que c'est l'initialisation intermédiaire qui donne les meilleurs résultats. Les deux estimations de paramètres donnent des résultats relativement corrects en terme de scores de synchronisation et intra-bande :

- pour $INI = 0.5$, on obtient comme rangs moyens : $intra \simeq 2.3$, $synchro \simeq 4.0$, $segmentation \simeq 5.1$
- pour $INI = 1$, on a : $intra \simeq 2.8$, $synchro \simeq 3.5$, $segmentation \simeq 4.6$

L'importante différence d'efficacité vient du fait que les coefficients estimés sont sensiblement différents en ce qui concerne les coefficients de synchronisation et d'attache aux données : on ne se trouve pas dans le même maximum local de notre fonction "qualité"... Les figures suivantes montrent à quel point les coefficients sont différents :on observe notamment des coefficients strictement négatifs en partant de $INI = 0.5$, ce qui n'est pas du tout le cas avec l'autre initialisation.

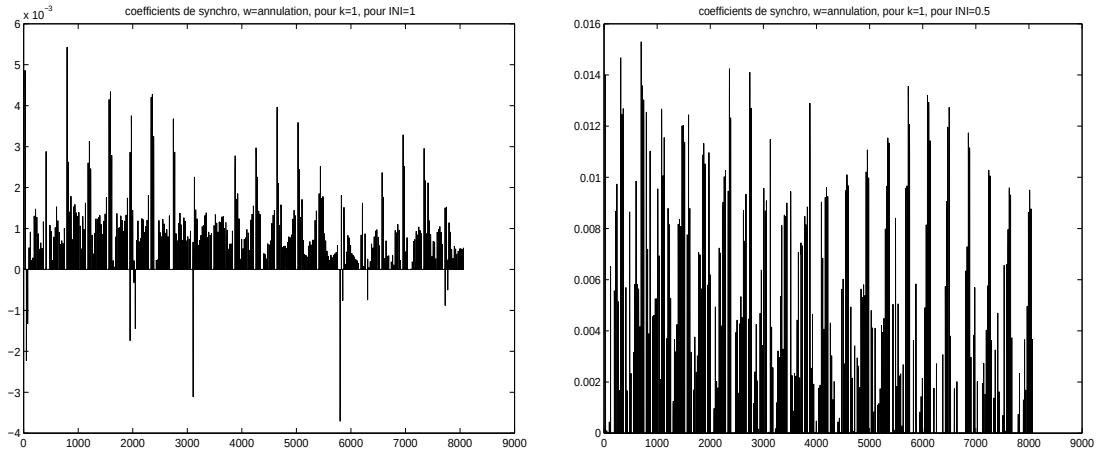

FIG. 10 – Les coefficients de synchronisation pour la première bande du mot "annulation" pour 2 initialisations différentes

Conclusion pour les initialisations L'initialisation des paramètres intra-bandes est valide. Par contre, celle des paramètres de synchronisation n'est pas optimale lorsque on lui applique la même méthode : mieux vaut ne pas présumer des valeurs des paramètres de synchronisation et les mettre à zéro. Cette initialisation conduit en effet à des performances moins bonnes après une longue estimation des paramètres qu'avec l'utilisation directe d'une initialisation d'un autre type !

Et si $\beta \neq 2$? Dans le cas $\beta = 1$, on obtient, pour de faibles valeurs du nombre d'itérations de l'EM (5) :

INI :	0	1
Reconnaissance :	$94.8\ \%$	$97.8\ \%$

Ces résultats qui ne semblent pas si mauvais à priori, cachent en fait une très mauvaise estimation des paramètres du modèle caché : les rangs moyens des 2 avoisinent les 5.5, ce qui est très mauvais. La problème de l'initialisation des paramètres de synchronisation pour les valeurs de $\beta \neq 2$ reste par conséquent encore à résoudre : il faudrait trouver mieux qu'une mise à zéro des coefficients du modèle.

6.3 l'algorithme EM

Nous ne prétendrons pas faire ici une étude générale de la convergence de l'algorithme d'estimation des paramètres : en effet, celle-ci dépend fortement de son initialisation. En revanche, nous allons donner une idée de sa vitesse de convergence pour $\beta = 1$ et pour l'initialisation la plus convenable : $INI = 0.5$.

Le gradient stochastique Dans notre programme a été implémenté un algorithme permettant de calculer pour les différents types de paramètres différentes grandeurs :

- la valeur moyenne de l'écart relatif entre les paramètres de l'étape n à l'étape $n+1$
- la valeur maximale de cet écart relatif

On peut avoir ainsi une idée de la vitesse de convergence du gradient stochastique. Les figures suivantes montrent l'évolution de ces 2 grandeurs pour les 2 types de coefficients relatifs au modèle caché, lors du premier gradient stochastique (mot annulation) :

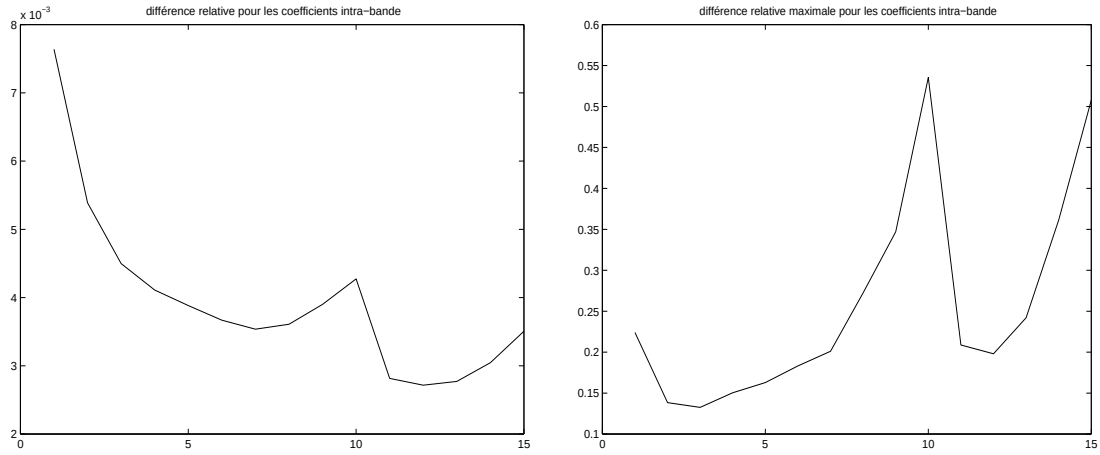

FIG. 11 – Une évolution en moyenne des coefficients intra-bande durant un gradient stochastique

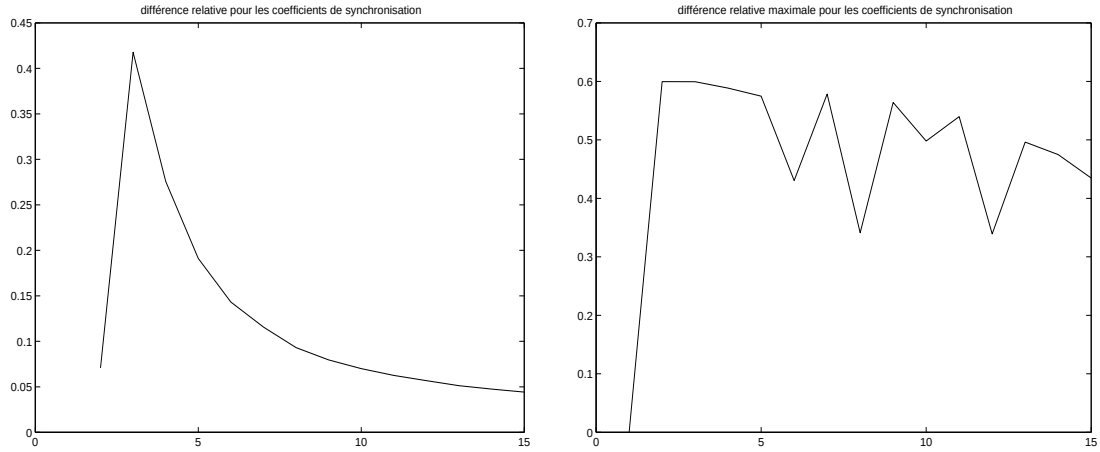

FIG. 12 – L'évolution homologue pour les coefficients de synchronisation

Interprétation Les résultats obtenus ici sont représentatifs des algorithmes de gradient stochastique implémentés ici. On peut tirer de ces résultats les enseignements suivants :

1. en moyenne, la convergence est assez rapide, et 5 itérations suffisent pour avoir une variation relative de moins de 5% pour les deux types de coefficients
2. la convergence des coefficients intra-bande est de loin la meilleure : elle est 10 fois plus rapide que pour les coefficients de synchronisation !
3. même après un nombre d'itérations assez important (15), certains coefficients de synchronisation continuent à connaître de fortes variations : leur convergence est très lente pour la norme "max".

On voit ainsi que l'estimation des paramètres intra-bande est bien plus facile que celle des coefficients de synchronisation, qui peinent à tous se stabiliser. Mais on a vu également que 5 itérations suffisaient pour obtenir une variation relative moyenne inférieure à 20%, pour un nombre d'itérations faible. C'est la raison pour laquelle on a conservé en pratique ce nombre d'itérations pour l'algorithme de gradient.

Convergence des moyennes et des variances Intéressons-nous à présent à la convergence des moyennes et des variances caractéristiques de chaque état caché. Les figures suivantes montrent les résultats obtenus pour 15 itérations de l'algorithme EM, toujours pour le mot "annulation".

Analyse des résultats Voici ce que l'on peut dire de la convergence des paramètres d'attache aux données par EM :

1. la convergence en moyenne est rapide : au bout de 5 itérations, la variation moyenne ne dépasse pas 2% dans les 2 cas
2. la convergence des moyennes est nettement plus rapide (10 fois plus !) que celle des variances
3. certaines variances peuvent encore subir des variations de l'ordre de 10% au bout de 15 itérations : la convergence au sens du max est relativement lente.

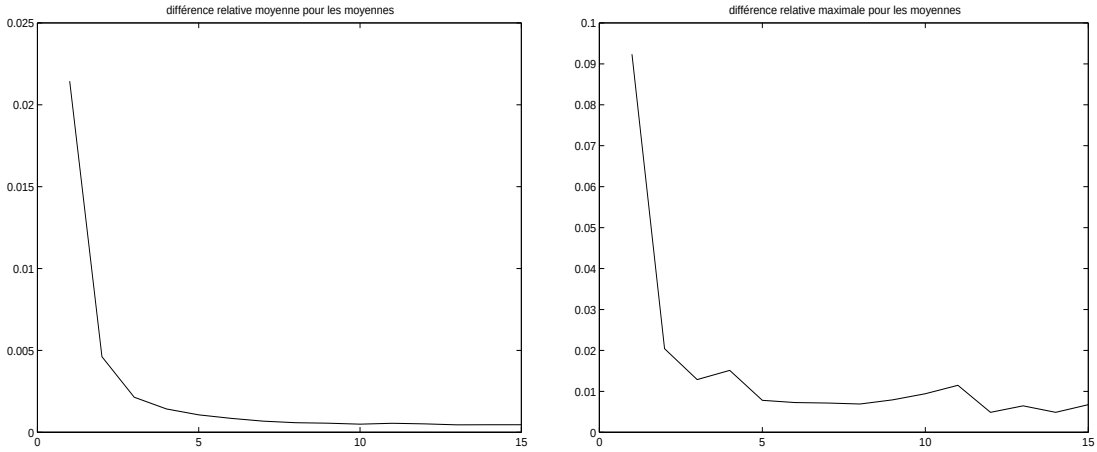

FIG. 13 – Une évolution en moyenne des moyennes durant l'algorithme EM

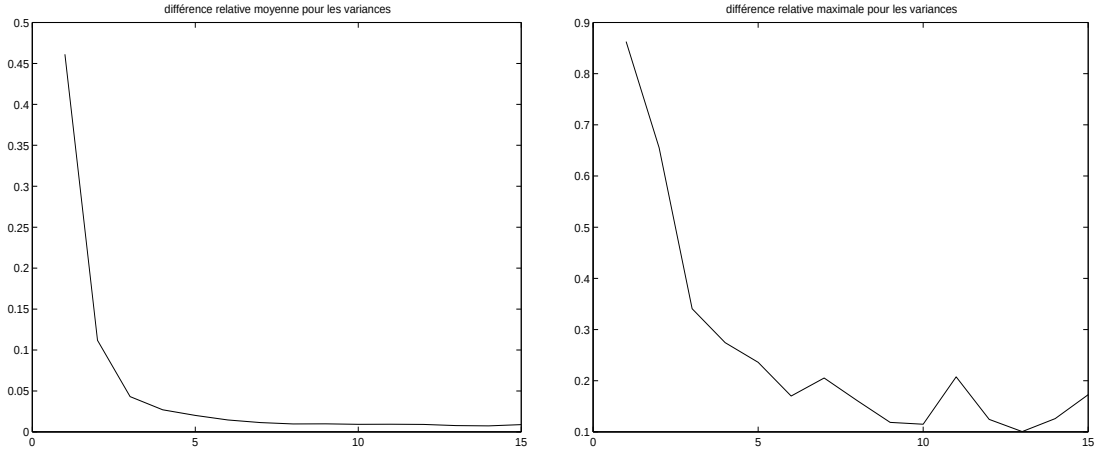

FIG. 14 – L'évolution correspondante pour les variances

Conclusion au sujet de l'EM L'estimation des paramètres via l'algorithme EM implémenté donne des résultats assez satisfaisant au sens de la convergence en moyenne (ou $\mathcal{L}^1$), pour 5 itérations pour chacun des gradients stochastiques et plus de 5 itérations d'EM. Cependant, la convergence ne s'effectue pas à la même vitesse pour tous les paramètres : seule l'estimation des moyennes des états cachés est très rapide. L'estimation des variances est nettement plus lente, et des fluctuations non-négligeables persistent après 15 itérations. C'est pourquoi un nombre plus important d'itérations apporte un surcroît de précision dans l'estimation. En ce qui concerne les paramètres intra-bande, on voit que l'estimation de ceux-ci est de relativement bonne qualité, même si l'on observe là aussi d'occasionnelles fluctuations. Par contre, la grande difficulté de l'estimation des paramètres réside dans le traitement des coefficients de synchronisation, très lents à se stabiliser, ce qui rend les caculs précis très longs.

6.4 La connexité verticale

Trop de coefficients de synchronisation ? Dans le modèle général que nous avons présenté, les coefficients de synchronisations $(F_i^{k,l})$ peuvent a priori être tous non-nuls. En partique, cela pose de gros problèmes : leur estimation est extrêment lente. En effet, pour 10 itérations d'EM avec 5 itérations de gradient stochastique, on a atteint un taux de reconnaissance très faible : $69,2\%$, avec un rang moyen pour le score de synchronisation de 5.2 : ils sont très mal estimés ! !
Comment résoudre ce problème sans faire travailler durant des semaines des machines surpuissantes sur un petit dictionnaire de 10 mots ? Une idée simple consiste à réduire le nombre de ces coefficients de synchronisation. En effet, ceux-ci sont de toute façon trop nombreux : pour $N = 10$, il y en a : $24^2 * 10 = 5760$! ! Or, notre signal ne comporte qu'un millier de coefficients ! Ce nombre est par conséquent bien trop élevé. Pour le réduire tout en conservant un couplage entre toutes les bandes, on peut se baser sur une notion que nous avaons mentionnée, sans la définir, auparavant : la *connexité verticale*. Elle repose sur le concept de voisinage : pour le modèle général, le voisinage "vertical" d'un état était constitué des tous les états des autres bandes de fréquences : tous les $(F_i^{k,l})$ pouvaient être non-nuls. On peut décider de réduire ce voisinages aux plus proches voisins uniquement : on dira que l'on adopté une "$2 * n$ connexité verticale" lorsque pour tous les k et l vérifiant $|k - l| > n$, $F_i^{k,l} = 0$. Autrement dit, on ne conserve les couplages qu'entre plus proches voisins.

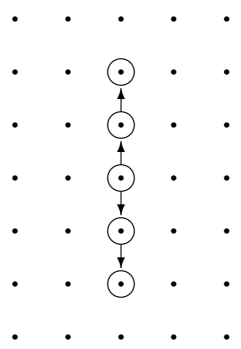

FIG. 15 – La 4-connexité verticale

Choix d'une connexité verticale Afin de déterminer une connexité verticale convenable, une série de tests a été effectuée. Toutes les expériences ont été menées avec 1 itération pour les gradients stochastiques, et au total 10 itérations d'EM et avec $INI = 1$. On a :

connexité verticale :	1	2	3	5	24
Reconnaissance :	98 %	95 %	84 %	59.6 %	69.2 %

Par conséquent, deux choix de connexité verticale sont très intéressants : la 2-connexité et la 4-connexité verticales. Pour conserver un couplage assez fort entre les bandes, il a été décider de se restrindre à la 4-connexité verticale dans les expérimentations. Ce choix s'est révélé payant,

puisque le taux de reconnaissance le plus élevé, toutes expériences confondues, a été obtenu en 4-connexité verticale :

$\boxed{98,8\%}$ de reconnaissance avec INI=0.5, 5 itérations de gradient de 40 itérations d'EM

6.5 La fonction de partition

Utilisation du calcul exact Comme nous l'avons vu à la fin de la $5^{\text{ième}}$ partie, il est possible de calculer de manière exacte les fonctions de partition $Z_{\underline{A}}$ dans le cas où les probabilités de transitions p_i sont 2 à 2 distinctes. Ce cas de figure est-il fréquent ? Généralement, le nombre de probabilités distantes de moins de 10^{-7} ne dépasse pas la dizaine. Cependant, les erreurs d'arrondi dans les calculs numériques rendent inutilisables le calcul exacte dans bon nombre de cas ! On obtient en effet une fonction de partition supérieure à 1 ou bien négative, alors que c'est une probabilité ! Tel est le cas pour les 7 plus longs mots du dictionnaire !
Durant l'étape de reconnaissance, on a implémenté un algorithme permettant de détecter et garder en mémoire les mots et les bandes posant un tel type de problème pour le calcul de la fonction de partition. Une fois qu'une bande est jugée impropre au calcul exact, c'est l'approximation classique qui est utilisée. Cette démarche a donné les résultats suivants :

nombre d'itérations pour l'EM :	Z_0	$Z_{\underline{A}}$
5 :	$92.8\ \%$	$93.8\ \%$
10 :	$95.0\ \%$	$95.4\ \%$
15 :	$93.8\ \%$	$95.4\ \%$

(résultats pour 1 itération de gradient stochastique et $INI = 1$). On voit ainsi que l'utilisation *ad hoc* du calcul exact donne de meilleurs résultats. Cependant, cette performance est simplement dûe à la moindre importance de la fonction de partition, plus petite pour le calcul exact. Ce procédé "compense" ainsi simplement la mauvaise estimation des paramètres de synchronisation !
D'autres expériences numériques ont montré que lorsque l'estimation se fait plus fine, l'utilisation de ce procédé n'apporte pas de meilleures performances, voire diminue l'efficacité du logiciel. Ainsi, le calcul théorique exacte s'est révèlé pratiquement inutilisable !

6.6 Résultats sur des signaux bruités

Protocole expérimental Une des grandes difficultés rencontrées en reconnaissance consiste à tenter d'identifier des signaux peu ou fortement bruités. Les performances des logiciels existant baissent très rapidement. Afin de tester les performances du logiciel en présence de bruit, nous avons ajouté deux types de bruits aux signaux de test.

Bruit blanc Dans un premier temps, intéressons-nous aux performances du modèle pour des siganux bruités uniformément en fréquence. Des expériences ont été menées pour trois niveaux de bruit différents : 10, 20 ou 30 dB de rapport signal sur bruit. Ces niveaux de bruitage correspondent à un fort bruit, un bruit moyen, et un faible bruit (auditivement parlant). Les paramètres du modèle ont été appris de trois façons différentes :

1. Cas 1 : 40 itérations d'EM, avec 5 itérations pour le gradient stochastique, $INI = 0.5$
2. Cas 2 : mêmes réglages, mais avec 80 itérations d'EM

3. Cas 3 : choix identiques qu'en 1, à ceci près que l'on est en 0-connexité verticale : on ne fait pas intervenir de synchronisation entre bandes.

Ces 3 cas de figure vont permettre d'évaluler l'efficacité du logciel, ainsi que les performances apportées par la synchronisation. On obtient :

RSB (dB) :	30	20	10
Cas 1	68%	15.2%	10%
Cas 2	66.8%	15.6%	10%
Cas 3	65.8%	16%	10%

Ansi, pour un bruit blanc, les performances du logiciel sont assez médiocres, notamment à 30 dB de RSB (bruit très faible). On constate un léger mieux pour le modèle avec synchronisation, surtout à 30 dB : on a 2% de reconnaissance supplémentaire.

Bruit coloré Nous avons également ajouté aux signaux de test un bruit coloré "grave", au sens où il a été fitré comme suit :

$$B_{\text{coloré}}[n] = B_{\text{blanc}} \star F_{AR}\,[n]$$

où $F_{AR}[n]$ est le filtre auto-régressif définit par sa transformée en z : $\hat{F}_{AR}(z) = \frac{1}{1-az^{-1}}$ avec $a = 0.95$. On a donc appliqué au bruit un filtre passe-bas. Toujours avec les mêmes réglages, l'expérience donne :

RSB (dB) :	30	20	10
Cas 1	90%	40.8%	10%
Cas 2	91.2%	41.2%	10%
Cas 3	89.6%	39.6%	10%

Les performances sont globalement meilleures, ce qui est normal, puisque le bruit est plus faible. On voit ici à nouveau le léger mieux apporté par la synchronisation : le taux de reconnaissance esst plus élevé de 1 à 2 %.
Cependant, cela est loin d'être satisfaisant : à l'oreille, les mots restent très nettement identifiables, même avec un rapport signal sur bruit de 10 dB. N'ayant pas pu, faute de temps, effectuer des tests pour de nombreux réglages, nous n'avons pas pu déterminer si il existe des réglages nettement plus performants pour notre modèle. Nous contenterons donc d'une conclusion en demie-teinte pour la reconnaissance appliquée à des signaux bruités : la synchronisation est utile, mais elle n'apporte qu'un léger mieux au taux de reconnaissance qui reste faible dans le cadre de notre étude.

Conclusion

Au cours du stage, la modélisation par champ de Markov du signal de parole a pu être intégralement mise en œuvre dans un logiciel complet de reconnaissance vocale programmé en C. On ainsi pu être étudié de nombreux problèmes, à savoir :

- la complexité de la phase d'estimation des paramètres du modèle : avec notamment le problème de l'initialisation, du choix d'approximations pertinentes, de la vitesse de convergence de l'algorithme
- le calcul détaillé du score acoustique de chacun des mots du dictionnaire pour un signal donné, équivalent à un problème de calcul de fonction de partition
- l'optimisation du modèle

Du point de vue théorique, plusieurs contributions originales auront été apportées : une analyse des propriétés fondamentales de l'espace contraint, une méthode de calcul systématique de grandeurs statistiques dans l'espace contraint uniforme, et un calcul exact de fonction de partition, également exploité dans la pratique.
Pratiquement parlant, un grand nombre d'expériences numériques auront été menées, permettant de mettre en évidence une bonne initialisation des paramètres du modèle, un ordre de grandeur de coefficients de synchronisation à conserver, ainsi que le comportement du modèle en présence de bruit.

Cette étude, qui s'inscrit dans le prolongement de la thèse de Guillaume Gravier [6], aura permis de valider à nouveau l'utilisation des champs de Markov en parole, puisque le modèle présente un taux de reconnaissance correct pour des signaux de bonne qualité. Il présente même un taux de reconnaissance légèrement supérieur au modèle dépourvu de synchronisation, en présence de bruit, ce qui est prometteur.

A la suite de ces travaux, un grand nombre d'axes de recherche restent encore à explorer dans le cadre d'un modèle multibande et segmental de parole avec synchronisation interbande. En voici quelques uns :

- l'optimisation de l'estimation des paramètres du modèle
- l'estimation des paramètres de synchronisation, médiocre pour l'instant
- la modélisation de l'observation (Y_t^k) conditionnellement à la connaissance de la segmentation cachée
- la forme du potentiel de synchronisation est-elle adéquate ?
- peut-on mieux estimer le score acoustique d'un mot ?

De nouvelles réponses à ces questions permettront sans doute à l'avenir d'accroître fortement les performances du modèle, qui n'en est qu'au tout début de son développement.

Références

[1] ? ? ? ? ? ? *Pour la Science,* ? ? 2001.

[2] ? ? ? ? ? ? *Pour la Science,* ? ? 2001.

[3] Edouard BRÉZIN. *Cours de Physique Statistique*. Editions de l'Ecole Polytechnique, 1999.

[4] Eva SYKOVÁ Charles NICHOLSON. Extracellular space structure revealed by diffusion analysis. *Trends Neurosci.,* (21) :207–215, 1998.

[5] L. R. RABINER et B-H JUANG. *Fudamentals of Speech Recognition*. Editions Prentice-Hall, 1999.

[6] Marc SIGELLE et Florence TUPIN. *Champs de Markov en Traitement d'Image*. Cours de l'Ecole Nationale Supérieure des Télécommunications de Paris, 1999.

[7] Jean-Louis BASDEVANT et Jean DALIBARD. *Mécanique Quantique*. Editions de l'Ecole Polytechnique, 1998.

[8] Jacques NEVEU et Nicole EL KAROUI. *Probabilités*. Editions de l'Ecole Polytechnique, 1998.

[9] Bernard LARROUTUROU et Pierre-Louis LIONS. *Méthodes mathématiques pour les Sciences de l'Ingénieur : Optimisation et Analyse numérique*. Editions de l'Ecole Polytechnique, 1998.

[10] Guillaume GRAVIER. *Analyse statistique à deux Dimensions pour la Modélisation segmentale du Signal de Parole - Application à la Reconnaissance*. Télécom Paris, Thèse de Doctorat, 2000.

[11] Marc SIGELLE Isabelle BLOCH, Henri MAITRE. *Introduction à l'Image*. Département Traitement du Signal et de l'Image, Ecole Nationale Supérieure des Télécommunications, 2000.

[12] B. KASTLER. *Principes de l'IRM. Manuel d'Auto-Apprentissage*. Masson, 1994.

[13] Patrick LALAURIE. Le centre international des sports de paris-bercy. Annales de l'institut technique du batiment et des travaux publics, Association française des Ponts et Chaussées, juin 1985.

[14] Denis LE BIHAN, editor. *Diffusion and Perfusion Magnetic Resonance Imaging*. Raven Press, Ltd, 1995.

[15] Patrick LE TALLEC. *Introduction à la Dynamique des Strucutures, cours de Majeure SICS*. Presses de l'Ecole Polytechnique, 1999.

[16] Stéphane MALLAT. *Traitement du Signal*. Editions de l'Ecole Polytechnique, 1999.

[17] Stéphane MALLAT. *Une Exploration des Signaux en Ondelettes*. Les Editions de l'Ecole polytechnique, 2000.

[18] Eric MOULINES. *traitement statistique du Signal et de l'Image*. Département Traitement du Signal et de l'Image, Ecole Nationale Supérieure des Télécommunications, 1998.

[19] D. LE BIHAN P. J. BASSER, J. MATTIELLO. Diagonal and off-diagonal components of the self-diffusion tensor ; their relation to and estimation from the mnr spin-echo signal. *Proc. 11th Annu. Meet. SMRM, Berlin,* (1) :1222, 1992.

[20] PierreBRÉMAUD. *Markov Chains, Gibs Fields, Monte Carlo Simulation, and Queues*. Springer, 1999.

[21] Cyril POUPON. *Détection des faisceaux de fibres de la substance blanche pour l'étude de la connectivité anatomique cérébrale*. PhD thesis, Ecole Nationale Supérieure des Télécommunications, décembre 1999.

[22] Walter RUDIN. *Analyse réelle et complexe, 3^e édition*. Dunod, 1998.

[23] Jean SALENÇON. *Mécanique des milieux continus, Tronc Commun, tomes I, II et III*. Presses de l'Ecole Polytechnique, 1998.

[24] Jean SALENÇON. *Plasticité et calcul à la rupture, cours de Majeure de Mécanique*. Presses de l'Ecole Polytechnique, 1999.

A Appendice : exemples de résultats de tests

A.1 4-connexité, 10 itérations d'EM, 4 itérations de gradient stochastique, $INI = 1$, $\beta = 2$

```
       Detailled statistics for annulation :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  10, intra:  50 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:  80, intra:  30 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  10, intra:   0 per, synchro  1
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  20 per, synchro  9
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:  40 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:  60 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 9.6, duree: 2.0, intra: 1.9, synchro 3.9.


       Detailled statistics for casino :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  60, intra:  30 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:  50 per, duree:   0, intra:  40 per, synchro 10
rank  3 :   0 per, data:   0 per, segmentation:  10 per, duree:   0, intra:  10 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:  40 per, duree:  20, intra:   0 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  10 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:  10, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:  10, intra:  10 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 2.9, duree: 2.7, intra: 2.6, synchro 2.0.


       Detailled statistics for cinema :
rank  1 :  88 per, data:  94 per, segmentation:   0 per, duree:  30, intra:  10 per, synchro
rank  2 :  12 per, data:   6 per, segmentation:   0 per, duree:  20, intra:  30 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  14, intra:  16 per, synchro  2
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:  18, intra:  26 per, synchro  5
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  10 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:  10 per, duree:   0, intra:   8 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:  38 per, duree:  18, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:  38 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:  14 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.1, data: 1.1, segmentation: 7.6, duree: 3.1, intra: 3.2, synchro 4.0.


       Detailled statistics for concert :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  32, intra:  54 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:  42 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  24, intra:   4 per, synchro
```

```
rank  4 :   0 per, data:   0 per, segmentation:   4 per, duree:  20, intra:   0 per, synchro  2
rank  5 :   0 per, data:   0 per, segmentation:  20 per, duree:  10, intra:   0 per, synchro  2
rank  6 :   0 per, data:   0 per, segmentation:  32 per, duree:   0, intra:   0 per, synchro  1
rank  7 :   0 per, data:   0 per, segmentation:  40 per, duree:   8, intra:   0 per, synchro  1
rank  8 :   0 per, data:   0 per, segmentation:   4 per, duree:   0, intra:   0 per, synchro  1
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro  1
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 6.2, duree: 3.2, intra: 1.5, synchro 6.3.


       Detailled statistics for corso :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  56, intra:  70 per, synchro  4
rank  2 :   0 per, data:   0 per, segmentation:  82 per, duree:  18, intra:  14 per, synchro  5
rank  3 :   0 per, data:   0 per, segmentation:  14 per, duree:  22, intra:   0 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:   4 per, duree:   0, intra:   6 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:   2 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   6 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 2.2, duree: 1.8, intra: 1.9, synchro 1.6.


       Detailled statistics for guide :
rank  1 :  94 per, data:  86 per, segmentation: 100 per, duree:   4, intra:  18 per, synchro 10
rank  2 :   6 per, data:  10 per, segmentation:   0 per, duree:   4, intra:  20 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  30, intra:   6 per, synchro
rank  4 :   0 per, data:   2 per, segmentation:   0 per, duree:  44, intra:   8 per, synchro
rank  5 :   0 per, data:   2 per, segmentation:   0 per, duree:  14, intra:   4 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:  10 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  12 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   6 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   4 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  12 per, synchro

On average: rank: 1.1, data: 1.2, segmentation: 1.0, duree: 3.7, intra: 4.8, synchro 1.0.


       Detailled statistics for message :
rank  1 :  98 per, data: 100 per, segmentation:   0 per, duree:  44, intra:  30 per, synchro
rank  2 :   2 per, data:   0 per, segmentation:   0 per, duree:  18, intra:  30 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   4 per, duree:  28, intra:  34 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:  14 per, duree:   0, intra:   6 per, synchro  2
rank  5 :   0 per, data:   0 per, segmentation:  34 per, duree:   4, intra:   0 per, synchro  4
rank  6 :   0 per, data:   0 per, segmentation:  48 per, duree:   2, intra:   0 per, synchro  1
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro  1
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 5.3, duree: 2.3, intra: 2.2, synchro 5.0.
```

```
       Detailled statistics for musee :
rank  1 :  90 per, data:  94 per, segmentation:   0 per, duree:   0, intra:  44 per, synchro
rank  2 :   8 per, data:   6 per, segmentation:  88 per, duree:  20, intra:  24 per, synchro  4
rank  3 :   2 per, data:   0 per, segmentation:  10 per, duree:  22, intra:  10 per, synchro  3
rank  4 :   0 per, data:   0 per, segmentation:   2 per, duree:  48, intra:  10 per, synchro  1
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   8, intra:   2 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   6 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro

On average: rank: 1.1, data: 1.1, segmentation: 2.1, duree: 3.5, intra: 2.6, synchro 3.0.


       Detailled statistics for quitter :
rank  1 :  84 per, data: 100 per, segmentation:   0 per, duree:   0, intra:  12 per, synchro
rank  2 :  16 per, data:   0 per, segmentation:  62 per, duree:  72, intra:  38 per, synchro  5
rank  3 :   0 per, data:   0 per, segmentation:  38 per, duree:  10, intra:  20 per, synchro  3
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   2 per, synchro  1
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:  14, intra:  14 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   6 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   4 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro

On average: rank: 1.2, data: 1.0, segmentation: 2.4, duree: 2.6, intra: 3.3, synchro 2.9.


       Detailled statistics for suivant :
rank  1 :  96 per, data: 100 per, segmentation:   0 per, duree:  50, intra:  32 per, synchro
rank  2 :   4 per, data:   0 per, segmentation:   0 per, duree:   6, intra:  18 per, synchro  1
rank  3 :   0 per, data:   0 per, segmentation:  18 per, duree:  12, intra:  20 per, synchro  3
rank  4 :   0 per, data:   0 per, segmentation:  46 per, duree:   4, intra:  12 per, synchro  4
rank  5 :   0 per, data:   0 per, segmentation:  36 per, duree:   8, intra:   8 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:  20, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   6 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 4.2, duree: 2.7, intra: 2.9, synchro 3.7.

 On average: data: 1.0, segm: 4.3, duree: 2.8, intra: 2.7, synchro: 3.3.


 Total percentage of recognised words: 95.0
```

A.2 4-connexité, 40 itérations d'EM, 5 itérations de gradient stochastique, $INI = 0.5$, $\beta = 2$

```
       Detailled statistics for annulation :
```

```
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  10, intra:  50 per, synchro  5
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:  80, intra:  10 per, synchro  5
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  10, intra:  30 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  10 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation: 100 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 10.0, duree: 2.0, intra: 2.1, synchro 1.5.


       Detailled statistics for casino :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  60, intra:  70 per, synchro 10
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  10 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:  20, intra:  10 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:  10, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:  10, intra:  10 per, synchro
rank  8 :   0 per, data:   0 per, segmentation: 100 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 8.0, duree: 2.7, intra: 2.1, synchro 1.0.


       Detailled statistics for cinema :
rank  1 :  98 per, data: 100 per, segmentation:   0 per, duree:  30, intra:  14 per, synchro
rank  2 :   2 per, data:   0 per, segmentation:   0 per, duree:  20, intra:  22 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  14, intra:  30 per, synchro  2
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:  18, intra:  16 per, synchro  5
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   6 per, synchro  1
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:  10 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:  18, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   8 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:  92 per, duree:   0, intra:   2 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 8.9, duree: 3.1, intra: 3.2, synchro 3.9.


       Detailled statistics for concert :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  32, intra:  64 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:  22 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  24, intra:  12 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:  10 per, duree:  20, intra:   2 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:  40 per, duree:  10, intra:   0 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:  32 per, duree:   0, intra:   0 per, synchro  1
rank  7 :   0 per, data:   0 per, segmentation:  18 per, duree:   8, intra:   0 per, synchro  1
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro  3
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro  2
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   0 per, synchro  1
```

```
On average: rank: 1.0, data: 1.0, segmentation: 5.6, duree: 3.2, intra: 1.5, synchro 7.7.


       Detailled statistics for corso :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  56, intra:  78 per, synchro  1
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:  18, intra:  14 per, synchro  8
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  22, intra:   0 per, synchro
rank  4 :   0 per, data:   0 per, segmentation: 100 per, duree:   0, intra:   6 per, synchro
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:   2 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 4.0, duree: 1.8, intra: 1.4, synchro 2.0.


       Detailled statistics for guide :
rank  1 :  98 per, data:  94 per, segmentation: 100 per, duree:   4, intra:  42 per, synchro
rank  2 :   2 per, data:   6 per, segmentation:   0 per, duree:   4, intra:  40 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  30, intra:   8 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:  44, intra:   0 per, synchro  2
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:  14, intra:   2 per, synchro  1
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:   2 per, synchro  2
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   4 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro  1

On average: rank: 1.0, data: 1.1, segmentation: 1.0, duree: 3.7, intra: 2.2, synchro 5.7.


       Detailled statistics for message :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  44, intra:  56 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:  18, intra:  32 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  28, intra:  12 per, synchro  2
rank  4 :   0 per, data:   0 per, segmentation:  70 per, duree:   0, intra:   0 per, synchro  4
rank  5 :   0 per, data:   0 per, segmentation:  30 per, duree:   4, intra:   0 per, synchro  2
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   0 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   4, intra:   0 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 4.3, duree: 2.3, intra: 1.6, synchro 4.1.


       Detailled statistics for musee :
rank  1 :  92 per, data:  92 per, segmentation:   0 per, duree:   0, intra:  60 per, synchro
rank  2 :   8 per, data:   8 per, segmentation:  40 per, duree:  20, intra:  14 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:  60 per, duree:  22, intra:  10 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:  48, intra:   0 per, synchro  1
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:   8, intra:   4 per, synchro  2
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro  2
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   2 per, synchro  2
```

```
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   6 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.1, data: 1.1, segmentation: 2.6, duree: 3.5, intra: 2.3, synchro 5.7.


       Detailled statistics for quitter :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:   0, intra:   4 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:  58 per, duree:  72, intra:  30 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:  42 per, duree:  10, intra:  30 per, synchro  4
rank  4 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:  18 per, synchro  3
rank  5 :   0 per, data:   0 per, segmentation:   0 per, duree:  14, intra:  10 per, synchro
rank  6 :   0 per, data:   0 per, segmentation:   0 per, duree:   2, intra:   4 per, synchro
rank  7 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 2.4, duree: 2.6, intra: 3.3, synchro 3.8.


       Detailled statistics for suivant :
rank  1 : 100 per, data: 100 per, segmentation:   0 per, duree:  50, intra:  18 per, synchro
rank  2 :   0 per, data:   0 per, segmentation:   0 per, duree:   6, intra:  24 per, synchro
rank  3 :   0 per, data:   0 per, segmentation:   0 per, duree:  12, intra:  28 per, synchro
rank  4 :   0 per, data:   0 per, segmentation:  42 per, duree:   4, intra:   8 per, synchro  2
rank  5 :   0 per, data:   0 per, segmentation:  52 per, duree:   8, intra:  10 per, synchro  5
rank  6 :   0 per, data:   0 per, segmentation:   4 per, duree:  20, intra:   8 per, synchro  1
rank  7 :   0 per, data:   0 per, segmentation:   2 per, duree:   0, intra:   2 per, synchro
rank  8 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   2 per, synchro
rank  9 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro
rank 10 :   0 per, data:   0 per, segmentation:   0 per, duree:   0, intra:   0 per, synchro

On average: rank: 1.0, data: 1.0, segmentation: 4.7, duree: 2.7, intra: 3.1, synchro 5.0.

 On average: data: 1.0, segm: 5.1, duree: 2.8, intra: 2.3, synchro: 4.0.


 Total percentage of recognised words: 98.8
```