Représentations Robustes pour la Reconnaissance Automatique de la Parole.

Alexey Ozerov

Stage de DESS CSA
Mars - Septembre 2003

Stage encadré par M. G. Gravier
Suivi par M. P. Fischer

Résumé

La représentation du signal par des coefficients cepstraux est souvent utilisée en Reconnaissance Automatique de la Parole (RAP). Bien que les coefficients cepstraux soient utilisés en raison de leurs bonnes propriétés de représentation, notamment la décorrélation des coefficients, ils souffrent de plusieurs limitations. En particulier ils sont sensibles aux conditions d'acquisition du signal et à l'environnement acoustique (problème de robustesse). A cause de cette sensibilité, la performance d'un système de reconnaissance de la parole est dégradée, elle est encore plus dégradée quand les conditions de l'apprentissage et de l'utilisation du système sont différentes. Le but de ce travail est d'étudier et de mettre en oeuvre des représentations robustes aux différences entre les conditions acoustiques d'apprentissage et d'évaluation. Ces représentations seront ensuite évaluées sur le système Sirocco de reconnaissance de la parole 'grand vocabulaire'. Une attention particulière sera prêtée aux méthodes de filtrage des trajectoires cepstrales, en particulier au filtrage passe-bas (lissage des trajectoires cepstrales). Quelques autres méthodes seront aussi considérées.

Abstract

The representation of speech signals by cepstral coefficients is very often used in Automatic Speech Recognition (ASR). While cepstral coefficients are used because of their good representation properties, namely the decorrelation of coefficients, they suffer some limitations. Particularly they are sensitive to signal acquisition and to acoustic environment (robustness problem). Because of this sensitivity, the performance of speech recognition systems are degraded, even more when the conditions of training and testing are different. The aim of this work is to study and to implement representations, which are robust to the mismatches between the acoustic conditions of training and evaluation. These representations will be then tested on the Sirocco large vocabulary speech recognition system. A particular attention will be paid to the methods of cepstral trajectories filtering, especially band-pass filtering (smoothing of cepstral trajectories). Some other methods will be also considered.

Table des matières

Remerciements

Le travail présenté dans ce rapport de stage a été effectué au sein de l'équipe METISS[1] de l'IRISA[2]. Je tiens à remercier Guillaume Gravier, chercheur à l'IRISA, qui m'a accueilli au sein de cet institut, encadré durant six mois, et permis de découvrir un domaine passionnant qui est la reconnaissance de la parole.

Je veux également remercier Mathieu Ben, thésard à l'équipe METISS, et Laurent Benaroya, ancien thésard[3], pour quelques conseils utiles.

1. Modélisation et Expérimentation pour le Traitement des Informations et des Signaux Sonores
2. Institut de Recherche en Informatique et Systèmes Aléatoires
3. Laurent Benaroya a soutenu sa thèse avec succès le 26 Juin 2003.

Chapitre 1

Introduction

Ce rapport commence par une brève introduction sur la reconnaissance automatique de la parole (chapitre 2). Ensuite, les principes de la modélisation en reconnaissance de la parole sont détaillés dans le chapitre 3. Une partie considérable de ce chapitre est consacrée à la modélisation probabiliste par des modèles de Markov cachés. Le chapitre 4 présente le sujet du stage. Il décrit le module d'analyse acoustique d'un système de la reconnaissance de la parole, introduit le problème de robustesse et présente des méthodes prétendant y remédier. Les détails de mise en oeuvre de ces méthodes et d'autres explications techniques sont donnés dans le chapitre 5. Enfin les résultats expérimentaux, la discussion et les perspectives avec la conclusion sont représentées aux chapitres 6, 7 et 8 respectivement.

Chapitre 2

Présentation générale

Dans ce chapitre nous allons dans un premier temps présenter le domaine de la reconnaissance automatique de la parole ainsi ses applications. Ensuite nous dirons quelques mots sur le projet Sirocoo et préciserons l'objectif de ce stage.

2.1 Reconnaissance automatique de la parole

Le problème de la reconnaissance automatique de la parole consiste à extraire, à l'aide d'un ordinateur, l'information lexicale contenue dans un signal de parole. Plus simplement, à partir de l'enregistrement d'une phrase prononcée, le but consiste à obtenir une version écrite de cette phrase. Donc on a un signal de parole à l'entrée d'un système de reconnaissance automatique de la parole et on a un texte correspondant à la sortie.

Il est possible d'imaginer beaucoup d'applications potentielles pour la reconnaissance automatique de la parole. La liste d'applications suivante n'est bien sûr pas exhaustive.

1. Les systèmes de la réponse vocale interactive, par exemple la réservation des vols.
2. La communication avec l'ordinateur pour les handicapés.
3. Les systèmes de dictée.
4. Les jeux.

En conséquence du grand nombre d'applications, il est naturel qu'il existe plusieurs types de systèmes de reconnaissance de la parole. Chaque système est adapté pour avoir la meilleure performance (taux de reconnaissance) dans le cadre de sa propre tâche. Selon le domaine d'application, les systèmes de reconnaissance de la parole peuvent être classifiés selon les critères suivants:

1. Le système *dépendant du locuteur* (optimisé pour un locuteur bien particulier) ou *independant du locuteur* (pouvant reconnaître n'importe quel utilisateur). Evidemment, les systèmes dépendant du locuteur ont la meilleure performance et ils peuvent être utilisés comme systèmes de dictée, mais pas pour la réservation des vols.
2. Le système de reconnaissance de la *parole continue*, des *mots isolés* (chaque mot est séparé l'un de l'autre par une pause importante) ou des *mots clés* (dans ce dernier cas la tâche consiste à reconnaître des mots appartenant un petit vocabulaire bien défini et de rejeter tous les autres mots).
3. La *taille du vocabulaire*: petite (100 mots), moyenne (5000 mots) ou grande (20000 mots).

Il y a aussi plusieurs approches de reconnaissance basées sur des méthodes différentes. On choisit le type d'approche en fonction de la tâche de reconnaissance.

Les premiers succès en reconnaissance de la parole ont été obtenus dans les années 70 à l'aide d'une méthode de *reconnaissance par comparaison à des exemples*. L'algorithme DTW [1, 2] (*Dynamic Time Warping*, en anglais) est dans le fond de cette méthode. Cette approche est plutôt utilisée pour la reconnaissance des mots isolés avec un petit vocabulaire.

Au début des années 80, l'utilisation des modèles de Markov cachés (*HMM pour "Hidden Markov Model"* en anglais) a permis de grands progrès [3] . En principe, cette nouvelle approche n'est qu'une extension statistique de la méthode déterministe DTW. L'utilisation des modèles HMM a permis aussi de passer aux méthodes de *reconnaissance par modélisation d'unités de parole*, permettant de modéliser des unités de parole de plus petite taille (typiquement les phonèmes), ce qui est fondamental pour construire des systèmes de reconnaissance de la parole 'grand vocabulaire'.

La troisième approche consiste à utiliser la combinaison des modèles HMM avec des réseaux de neurones artificiels (*ANN pour "Artificial Neural Network"* en anglais) [2].

2.2 Le projet Sirocco

L'objectif du projet *Sirocco* [1] est de developper un système de reconnaissance de la parole 'grand vocabulaire' (> 10000 mots) indépendant du locuteur.

Ce projet est réalisé en collaboration avec plusieurs laboratoires français: l'ENST [2] de Paris, l'IRISA, l'IRIT [3] de Toulouse, le LIA [4] et le LORIA [5] de Lorrain. Le logiciel correspondant est ouvert et il est distribué sous *GNU Public Licence agreement* [6].

Puisque Sirocco est un système de reconnaissance de la parole 'grand vocabulaire', le reste de ce document concernera surtout ce type de systèmes, utilisant l'approche par des modèles HMM.

2.3 Le stage

La représentation du signal par des coefficients mel-cepstraux est souvent utilisée en reconnaissance automatique de la parole. Cette représentation est introduite dans la section 4.1. Bien que les coefficients mel-cepstraux soient utilisés en raison de leurs bonnes propriétés de représentation, notamment la décorrélation des coefficients, ils souffrent de plusieurs limitations. En particulier ils sont sensibles aux conditions d'acquisition du signal et de l'environnement acoustique (problème de robustesse). A cause de cette sensibilité, la performance d'un système de reconnaissance de la parole est dégradée, elle est encore plus dégradée quand les conditions de l'entraînement et de l'utilisation du système sont différentes. C'est pour cela que dans les guides de systèmes dépendant du locuteur, il est souvent très conseillé de ne changer ni de microphone ni de lieu d'enregistrement après la phase d'apprentissage. Evidemment il n'est pas possible d'accomplir toutes ces précautions pour un système indépendant du locuteur.

Le but de ce stage est d'étudier, de mettre en oeuvre et enfin d'évaluer sur le système Sirocco des représentations robustes aux différences entre les conditions acoustiques d'entraînement et d'évaluation. Nous prêterons une attention particulière aux méthodes de filtrage des trajectoires cepstrales, en particulier au filtrage passe-bas (lissage des trajectoires cepstrales). Quelques autres méthodes seront aussi considérées.

1. http://www.irisa.fr/sirocco/
2. Ecole Nationale Supérieur des Télécommunications
3. Institut de Recherche en Informatique de Toulouse
4. Laboratoire d'Informatique de l'université d'Avignon
5. Laboratoire Lorrain de Recherche en Informatique et ses Applications
6. http://www.gnu.org/copyleft/gpl.html

Chapitre 3

La modélisation en reconnaissance automatique de la parole

Dans ce chapitre nous introduirons tout d'abord touts les modules composant un système de reconnaissance de la parole en décrivant son fonctionnement conjoint. Ensuite nous faisons une introduction en théorie des modèles de Markov cachés, qui est une partie essentielle de la modélisation en reconnaissance de la parole. Enfin nous décrirons chacun de ces modules en peu plus en détail.

3.1 Le principe

Un système de reconnaissance de la parole 'grand vocabulaire' est souvent décomposé en plusieurs modules, généralement au nombre de cinq.

1. Un *module d'analyse acoustique* et de traitement du signal qui transforme le signal de parole en une séquence de *vecteurs acoustiques* (*feature vectors* en anglais). Cette représentation par des vecteurs acoustiques doit être adaptée pour la reconnaissance, c'est-à-dire que ce module a pour but de garder dans les vecteurs acoustiques toute information lexicale et de supprimer toutes les autres informations, telles que la *variabilité intra et inter-locuteur*, les *bruits ambiants* etc.
2. Des *modèles acoustiques,* qui sont des modèles statistiques (des "petits" HMMs) des phonèmes. Ces modèles sont entraînés à partir d'une grande quantité de données de parole (par exemple, enregistrement de nombreuses phrases) contenant plusieurs fois les différentes unités de parole dans plusieurs contextes phonétiques différents.
3. Des *modèles lexicaux,* qui sont des modèles des mots de la langue. Les modèles les plus simples sont fournis par un dictionnaire phonétique; les plus complexes sont des véritables automates probabilistes, capables d'associer une probabilité à chaque prononciation possible d'un mot. Ces modèles possèdent toutes les prononciations possibles de chaque mot du dictionnaire.
4. Un *modèle de langage,* qui associe une probabilité à toute suite de mots présents dans le lexique. Ce modèle est entraîné sur une grande base de texte.
5. Un *module de décodage* (reconnaissance elle-même), qui a pour but de retrouver la phrase correspondant "le mieux" à la phrase prononcée parmi toutes les phrases possibles. Le décodage s'effectue à l'aide de tous les modèles déjà présentés, et il est très important que la recherche de cette "meilleure phrase" se passe à tous les niveaux simultanément, c'est-à-dire au niveaux *acoustique* (modèles acoustiques), *phonétique* (modèles lexicaux) et *sémantique* (modèle de langage). Une telle approche est beaucoup plus performante que de détecter d'abord des phonèmes, ensuite des mots, et enfin toute la phrase.

Le module d'analyse acoustique sera présenté dans le chapitre 4, puisque le sujet principal de ce travail consiste surtout à améliorer ce module. Sans rentrer dans les détails, l'analyse acoustique consiste en l'analyse des trames recouvrant de signal d'entrée. La taille typique d'une telle trame est entre 20 et 30 ms avec un décalage autour de 10 ms. Chaque trame x_t est transformée en un vecteur acoustique $O_t = (o_t^1, o_t^2, \ldots, o_t^N)^T$. A partir d'un signal de parole, on obtient donc une suite de vecteurs acoustiques $O = O_1 O_2 \ldots O_T$.

Connaissant cette suite d'observations (vecteurs acoustiques) O, on cherche, parmi l'ensemble des suites de mots susceptibles d'avoir été prononcés, celle qui est la plus probable (critère de *Maximum A Posteriori (MAP)*):

$$\hat{W} = \underset{W}{\operatorname{argmax}}\, P(W \mid O) \tag{3.1}$$

avec $W = w_1 \ldots w_i \ldots w_K$, w_i étant un mot du vocabulaire. Cette probabilité n'étant pas calculable directement, on utilise la règle de Bayes:

$$P(W \mid O) = \frac{p(O \mid W)P(W)}{p(O)} \tag{3.2}$$

$P(W)$ est *le score linguistique*. Il correspond à la ***probabilité a priori*** d'observer la séquence de mots W. Cette probabilité est donnée par le modèle de langage. $p(O \mid W)$ s'appelle *le score acoustique* et il est la "vraisemblance" (la valeur d'une densité de probabilité[1]) de O, sachant que la suite de mots W a été prononcée. Cette vraisemblance est déterminée par des modèles acoustiques.

Puisque la vraisemblance $p(O)$ dans (3.2) ne dépend pas de W, on a:

$$\hat{W} = \underset{W}{\operatorname{argmax}}\, p(O \mid W)P(W) \tag{3.3}$$

Un schéma explicatif est présenté dans la figure 3.1.

3.2 Modèles de Markov cachés

Cette partie a pour but de faire une introduction théorique des modèles HMMs et de présenter les algorithmes généraux pour mieux comprendre ensuite le fonctionnement d'un système de reconnaissance de la parole.

3.2.1 Définitions

Définition 3.2.1. Un ***modèle de Markov discret*** M est un *automate stochastique à nombre d'états fini* satisfaisant la propriété suivante: "la probabilité d'être dans un état à un instant donné ne dépend que des états visités avant", c'est-à-dire que si on fait les notations suivantes:

- $\mathcal{S} = \{1,2,\ldots,L\}$ l'ensemble des états de M[2],
- s_t l'état de M visité à l'instant $t \in \mathbb{N}^*$,

un modèle de Markov est alors paramétrisé en termes d'un ensemble de probabilités de transition

$$P\left(s_t = j \mid s_{t-1} = i, s_{t-2} = n, \ldots\right) \tag{3.4}$$

Maintenant on fait les deux hypothèses simplificatrices suivantes:

1. On suppose que le modèle M est d'ordre 1, c'est-à-dire que la probabilité (3.4) de passer à un état particulier j à l'instant t ne dépend que de l'état à l'instant $t-1$. Ceci revient alors à supposer que

1. Ensuite on va toujours noter n'importe quelle vraisemblance par p sans donner le nom explicite de la densité de probabilité, ce sera comprehénsible grâce au contexte.
2. Les états sont marqués par des chiffres pour la simplification de l'explication.

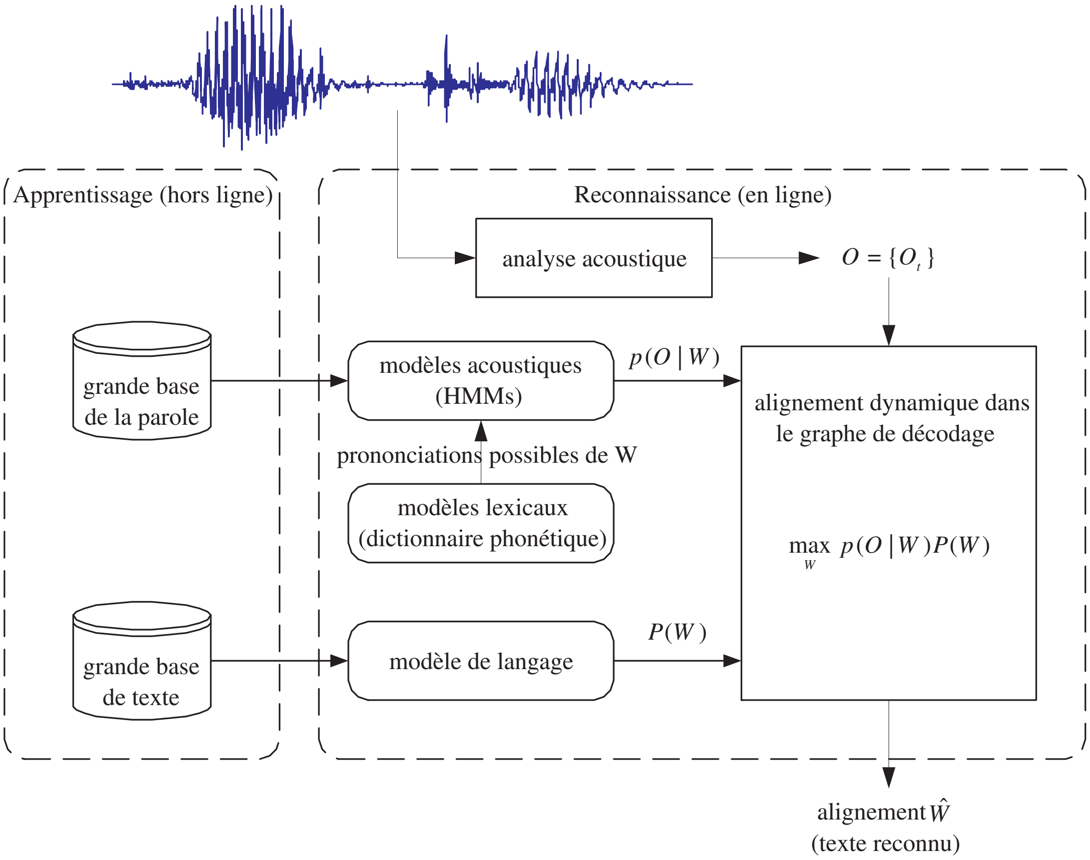

FIG. 3.1 – *Schéma général d'un système de reconnaissance de la parole.*

$$P\left(s_t = j \mid s_{t-1} = i, s_{t-2} = n, \ldots\right) = P\left(s_t = j \mid s_{t-1} = i\right) \tag{3.5}$$

2. On suppose également que ces probabilités de transition sont indépendantes du temps. Cela veut dire que

$$P\left(s_{t+1} = j \mid s_t = i\right) = a_{ij}, \qquad \forall t \in \mathbb{N}^* \tag{3.6}$$

Les probabilités a_{ij} sont appelées ***probabilités de transition*** et doivent (évidemment) vérifier les propriétés suivantes:

$$\begin{cases} a_{ij} \geq 0, & \forall i,j \\ \sum_{j=1}^{L} a_{ij} = 1, & \forall i \end{cases} \tag{3.7}$$

Un modèle de Markov discret est donc paramétrisé selon

- Son nombre d'états L.
- Sa matrice $(L \times L)$ de probabilités de transition:

$$A = \{a_{ij}\} \tag{3.8}$$

- La ***distribution de probabilités des états initiaux***:

$$\pi = (\pi_1, \pi_2, \ldots, \pi_L) \tag{3.9}$$

où $\pi_i = P(s_1 = i)$ est la probabilité d'être dans l'état i à l'instant $t = 1$.

L'ensemble des paramètres décrivant M est donc donné par $M = \{\pi, A\}$ [3].

Définition 3.2.2. Un ***modèle de Markov caché*** Λ (HMM pour *"Hidden Markov Model"*) est défini par un modèle de Markov discret dits ***modèle de base*** avec une loi de probabilité associée à chaque état. On a aussi une notion de ***séquence d'observations*** qui apparaît, c'est-à-dire qu'à chaque instant donné on observe une réalisation d'une variable aléatoire suivant la loi de probabilité associée à l'état visité à cet instant. Ces lois sont donc appelées les ***lois d'émission*** . Plus précisement, si on introduit les notations suivantes:

- M le modèle de base,
- $\mathcal{P} = \{P_\Lambda^1, P_\Lambda^2, \ldots, P_\Lambda^L\}$ l'ensemble des lois de probabilité associées aux états de M,
- s_t l'état de M visité à l'instant t,
- $O_t = \left(o_t^1, o_t^2, \ldots, o_t^N\right)^T$ (vecteur de dimension N) l'observation observée à l'instant t,

alors, sous la condition $s_t = i$, O_t est une réalisation d'une variable aléatoire ζ_t suivant la loi de probabilité P_Λ^i.

Cela revient à faire l'hypothèse supposant que les variables aléatoires ζ_t sont *indépendantes conditionnellement aux événements* $\{s_t = i\}$.

Dans la plupart des cas d'utilisation de ce modèle, on connaît la séquence d'observations et il faut retrouver la séquence d'états correspondante. C'est pourquoi ce modèle est dit "caché".

Un HMM est donc décrit par son modèle de base et par l'ensemble des lois de probabilité, c'est-à-dire que $\Lambda = \{\pi, A, \mathcal{P}\}$.

Ensuite les HMMs se distinguent par la typologie des lois de probabilité.

Définition 3.2.3. Un HMM est appelé un ***HMM à lois discrètes,*** si les lois de probabilité P_Λ^i sont discrètes et les variables aléatoires correspondantes ont des valeurs dans le même ensemble fini d'observations possibles. Cet ensemble est appelé parfois "alphabet". Si on note l'alphabet comme $\mathcal{A} = \{V_1, V_2, \ldots V_K\}$ (où K est la taille de l'alphabet), ces lois sont décrites par une matrice $(L \times K)$:

$$B = \{b_i(k)\} \tag{3.10}$$

avec $b_i(k) = P(O_t = V_k \mid s_t = i)$ la probabilité d'observer le symbole V_k à l'instant t, sachant que l'on est dans l'état i. Cette probabilité est généralement appelée ***probabilité d'émission.***

Ayant une séquence d'observations $O = O_1, O_2, \ldots, O_T$ et une séquence d'états correspondante $s = s_1, s_2, \ldots, s_T$, l'hypothèse d'indépendance, introduite dans la définition 3.2.2, se traduit par

$$\begin{split} P(O_1 = V_k, O_2 = V_l, \ldots, O_T = V_m \mid s_1 = i, s_2 = j, \ldots s_T = n) = \\ P(O_1 = V_k \mid s_1 = i)\, P(O_2 = V_l \mid s_2 = j) \ldots P(O_T = V_m \mid s_T = n) \end{split} \tag{3.11}$$

Un HMM à lois discrètes est donc paramétrisé par $\Lambda_D = \{\pi, A, B\}$.

Puisque dans la reconnaissance de la parole les observations (les vecteurs acoustiques) admettent un nombre de valeurs infini dans $\mathbb{R}^N$, la modélisation par des HMMs à lois discrètes ne peut pas s'effectuer directement. Elle est quand même parfois utilisée avec une procédure de quantification des vecteurs acoustiques. Mais les HMM à densités continues, qui vont être présentés ensuite, sont utilisés plus souvent.

Définition 3.2.4. Un HMM est appelé un ***HMM à densités continues*** , si les lois de probabilité P_Λ^i sont absolument continues sur $\mathbb{R}^N$. On a donc une densité continue sur $\mathbb{R}^N$ associée à

3. Pour ne pas alourdir les notations on va toujours noter le modèle et sa paramétrisation avec la même lettre.

chaque état i de M notée $f_i(\cdot)$. Maintenant l'observation O_t peut prendre n'importe quelle valeur dans $\mathbb{R}^N$, mais on ne peut plus calculer la probabilité d'émission, puisqu'elle est toujours nulle pour les lois continues. Donc on calcule la vraisemblance $p(O_t \mid s_t = i) \triangleq f_i(O_t)$, qui est appelée par analogie ***vraisemblance d'émission.*** [4]

L'hypothèse d'indépendance de la définition 3.2.2 s'écrit maintenant:

$$p\left(O_1,O_2,\ldots,O_T \mid s_1 = i,s_2 = j,\ldots s_T = n\right) = \\ p\left(O_1 \mid s_1 = i\right) p\left(O_2 \mid s_2 = j\right) \ldots p\left(O_T \mid s_T = n\right) \quad (3.12)$$

Ensuite on se pose la question du choix des densités continues pour la modélisation. Souvent, en première approximation, on choisit les *densités monogaussiennes multidimensionnelles*:

$$f_i(O_t) = \frac{1}{(2\pi)^{N/2} \left|\Sigma_i\right|^{1/2}} \exp\left(-\frac{1}{2}\left(O_t - \mu_i\right)^T \Sigma_i^{-1} \left(O_t - \mu_i\right)\right) \quad (3.13)$$

où μ_i et Σ_i représentent respectivement le vecteur moyen et la matrice de covariance associés à l'état i. Le modèle dans ce cas est paramétrisé ainsi:

$$\Lambda_C = \{\pi,A,\mu_i,\Sigma_i \mid i = 1,\ldots,L\} \quad (3.14)$$

Le choix d'une telle distribution est justifié par le *théorème limite central* et par le fait, que l'estimation des paramètres de cette distribution est beaucoup plus simple que pour les autres distributions. D'un autre côté ce choix est, quand même, assez restrictif: on ne peut pas approcher n'importe quelle distribution par une gaussienne. C'est pour cela que les *mélanges de gaussiennes* sont utilisés préférentiellement.

Définition 3.2.5. Un ***modèle de mélanges de gaussiennes*** (*GMM pour "Gaussian Mixture Model"* en anglais) peut être construit comme suit. Imaginons que les observations O sont générées par le mécanisme suivant:

1. On tire une variable aléatoire discrète η à valeurs dans $\{1,2,\ldots,K\}$ où K désigne le nombre de *composantes du mélange*, on note $c_k = P\{\eta = k\}$ pour $k = 1,2,\ldots,K$ les probabilités respectives de tirer chacune des composantes.
2. Conditionnellement à l'événement $\{\eta = k\}$, O est une réalisation de variable aléatoire, qui est distribuée selon la loi gaussienne multidimensionnelle $\mathcal{N}_N\left(\mu_k,\Sigma_k\right)$ dont la densité $g_k(O)$ est definie par (3.13).

On peut montrer, que O est une réalisation de variable aléatoire de densité:

$$g(O) = \sum_{k=1}^{K} P(\eta = k) p(O \mid \eta = k) = \sum_{k=1}^{K} c_k g_k(O) \quad (3.15)$$

Maintenant, on associe à chaque état i d'un HMM une densité de mélange de gaussiennes, qui s'écrit comme:

$$f_i(O_t) = \sum_{k=1}^{K} \frac{c_{ik}}{(2\pi)^{N/2} \left|\Sigma_{ik}\right|^{1/2}} \exp\left(-\frac{1}{2}\left(O_t - \mu_{ik}\right)^T \Sigma_{ik}^{-1} \left(O_t - \mu_{ik}\right)\right) \quad (3.16)$$

avec (bien évidemment) les contraintes:

4. La notion de vraisemblance utilisé ici est expliquée plus précisément dans l'annexe A.

$$\begin{cases} c_{ik} \geq 0, & \forall i,k \\ \sum_{k=1}^{K} c_{ik} = 1, & \forall i \end{cases} \tag{3.17}$$

Un HMM à densités continues est alors paramétrisé comme:

$$\Lambda_G = \{\pi, A, \mu_{ik}, \Sigma_{ik}, c_{ik} \mid i = 1, \ldots, L, k = 1, \ldots, K\} \tag{3.18}$$

Bien qu'un modèle de mélanges de gaussiennes soit décrit par un mécanisme très simple (voir la définition 3.2.5), sa distribution approche bien n'importe quelle autre distribution, si le nombre de composantes du mélange est suffisamment grand. Ces distributions sont alors largement utilisées en reconnaissance automatique de la parole.

Dans la suite de ce document on restera dans le cadre des densités monogaussiennes pour la raison de simplicité d'explication. Cela ne réduit pas beaucoup la généralité, puisqu'un modèle GMM peut être considéré comme un cas particulier de modèle HMM. Effectivement, un modèle GMM a aussi des états "cachés" (les composantes du mélange) et un processus sous-jacent (le tirage aléatoire décrit dans la définition 3.2.5). Un GMM à K composantes peut être alors remplacé par un HMM à K états et à densités monogaussiennes dont la matrice de transition a la forme suivante: $A = \{a_{lk}\} = \{c_k\}$, c'est-à-dire , que la probabilité d'entrer dans un certain état ne dépend pas de l'état précédent.

Pour donner une idée, un HMM à trois états est représenté dans la figure 3.2.

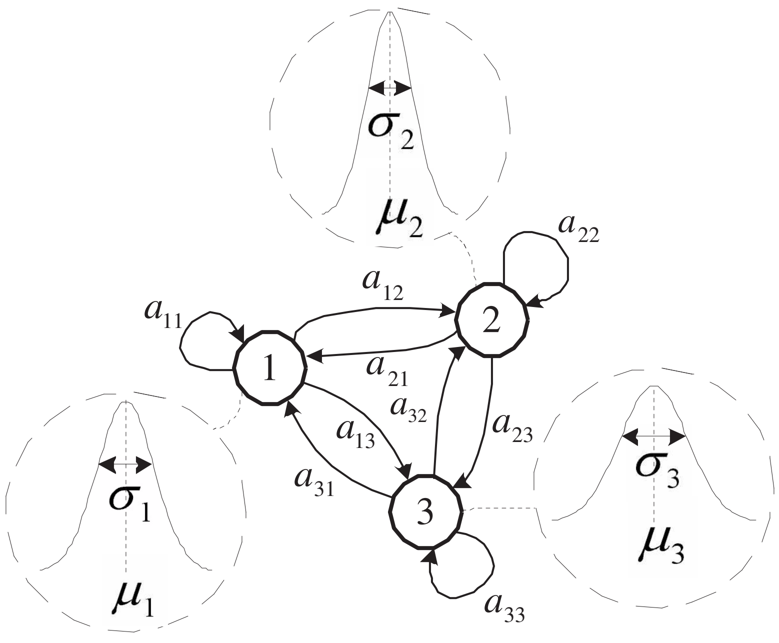

FIG. 3.2 – *HMM à densités continues monogaussiennes ($N = 1$, $L = 3$).*

3.2.2 Trois problèmes pour HMMs

Les tâches associées aux HMMs sont souvent formulées sous la forme de trois problèmes (voir [1] par exemple):

Problème 1 (Calcul de la vraisemblance d'une séquence d'observations). *Etant donné le modèle Λ, défini par (3.14), comment calcule-t-on $p(O \mid \Lambda)$, la vraisemblance de la séquence d'observations $O = O_1, O_2, \ldots, O_T$?*

Puisque l'ensemble de tous les événements $I = i_1,i_2,\ldots,i_T$ [5] possibles est une partition de l'espace probabiliste, on peut réécrire cette vraisemblance comme:

$$p(O \mid \Lambda) = \sum_I p(O \mid I,\Lambda)P(I \mid \Lambda) \tag{3.19}$$

où la somme est faite sur toutes les séquences d'états I possibles. En utilisant l'hypothèse d'indépendance (3.12), on a

$$p(O \mid I,\Lambda) = \prod_{t=1}^{T} p(O_t \mid s_t = i_t,\Lambda) = \prod_{t=1}^{T} f_{i_t}(O_t) \tag{3.20}$$

Ensuite, en utilisant la définition de la probabilité conditionnelle et l'hypothèse (3.5), on peut réécrire le deuxième terme de (3.19) comme:

$$\begin{aligned} P(I \mid \Lambda) &= P(s_1 = i_1 \mid \Lambda)\prod_{t=2}^{T} P(s_t = i_t \mid s_{t-1} = i_{t-1},\ldots,\Lambda) \\ &= P(s_1 = i_1 \mid \Lambda)\prod_{t=2}^{T} P(s_t = i_t \mid s_{t-1} = i_{t-1},\Lambda) \\ &= \pi_{i_1}\prod_{t=2}^{T} a_{i_t i_{t-1}} \end{aligned} \tag{3.21}$$

en déduisant enfin l'expression de la vraisemblance $p(O \mid \Lambda)$

$$p(O \mid \Lambda) = \sum_I \left[\pi_{i_1} f_{i_1}(O_1)\prod_{t=2}^{T} a_{i_t i_{t-1}} f_{i_t}(O_t)\right] \tag{3.22}$$

Pour calculer cette vraisemblance en utilisant directement l'équation (3.22), il faut effectuer $(2T-1)L^T$ multiplications (chaque terme de la somme demande $2T-1$ multiplications et il existe L^T séquences différentes d'états, c'est-à-dire L^T termes). En pratique c'est bien sûr impossible, même pour des valeurs de T assez petites, donc on voit bien que c'est un problème.

Problème 2 (Recherche de la séquence d'états optimale). *Etant donné le modèle Λ, comment choisir la séquence d'états $I = i_1,i_2,\ldots,i_T$ maximisant $p(O,I \mid \Lambda)$, la vraisemblance conjointe de la séquence d'observations $O = O_1,O_2,\ldots,O_T$ et de la séquence d'états? Donc on cherche I^*, tel que*

$$I^* = \operatorname*{argmax}_I p(O,I \mid \Lambda) \tag{3.23}$$

Selon (3.20) et (3.21) $p(O,I \mid \Lambda)$ s'écrit comme:

$$\begin{aligned} p(O,I \mid \Lambda) &= p(O \mid I,\Lambda)P(I \mid \Lambda) \\ &= \pi_{i_1} f_{i_1}(O_1)\prod_{t=2}^{T} a_{i_t i_{t-1}} f_{i_t}(O_t) \end{aligned} \tag{3.24}$$

5. Pour une séquence d'états fixée $I = i_1,i_2,\ldots,i_T$, on va parfois noter aussi par I l'évenement $\{s_1 = i_1,s_2 = i_2,\ldots,s_T = i_T\}$

Si on fait la recherche de I^* en utilisant directement l'expression (3.24), on voit bien qu'il faudra calculer cette expression pour chaque séquence d'états possible, c'est-à-dire L^T fois. On se retrouve alors avec la même complexité de calcul que pour le problème 1.

Problème 3 (Estimation des paramètres). *Comment ajuster les paramètres Λ du modèle HMM d'une façon telle que la vraisemblance $p(O \mid \Lambda)$ soit maximale? On cherche alors Λ^*, satisfaisant*

$$\Lambda^* = \underset{\Lambda}{\operatorname{argmax}}\, p(O \mid \Lambda) \tag{3.25}$$

3.2.3 Solution du problème 1: Procédure "avant-arrière"

Pour résoudre le problème 1, on introduit d'abord deux quantités:

$$\alpha_t(i) = p\left(O_1, O_2, \ldots, O_t, s_t = i \mid \Lambda\right) \tag{3.26}$$

la vraisemblance de la séquence d'observations partielle jusqu'à l'instant t et de l'état i à l'instant t, et

$$\beta_t(i) = p\left(O_{t+1}, O_{t+2}, \ldots, O_T \mid s_t = i, \Lambda\right) \tag{3.27}$$

la vraisemblance de la séquence d'observations partielle allant de $t+1$ jusqu'à T, sachant, que l'on était à l'état i à l'instant t.

La vraisemblance $p\left(O \mid \Lambda\right)$ peut se calculer à partir de ces deux quantités, donc on a deux façons de calculer cette vraisemblance:

Procédure "avant"

1. Initialisation, pour $1 \leq i \leq L$:

$$\alpha_1(i) = \pi_i f_i(O_1) \tag{3.28}$$

2. Recurrence "avant", pour $t = 1, 2, \ldots, T-1, 1 \leq j \leq L$:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{L} \alpha_t(i) a_{ij}\right] f_j(O_{t+1}) \tag{3.29}$$

3. Calcul de vraisemblance:

$$p\left(O \mid \Lambda\right) = \sum_{i=1}^{L} \alpha_T(i) \tag{3.30}$$

Procédure "arrière"

1. Initialisation, pour $1 \leq i \leq L$:

$$\beta_T(i) = 1 \tag{3.31}$$

2. Recurrence "arrière", pour $t = T-1, T-2, \ldots, 1, 1 \leq i \leq L$:

$$\beta_t(i) = \sum_{j=1}^{L} a_{ij} f_j(O_{t+1}) \beta_{t+1}(j) \tag{3.32}$$

3. Calcul de vraisemblance:

$$p(O \mid \Lambda) = \sum_{i=1}^{L} \pi_i f_i(O_1)\beta_1(i) \tag{3.33}$$

Dans l'équation (3.29) de la procédure "avant" on calcule tout d'abord $p(O_1, \dots ,O_T,s_{t+1} = j \mid \Lambda)$ la vraisemblance d'émettre la séquence d'observations partielle $O_1,O_2, \dots ,O_t$ et de passer à l'état j à l'instant $t+1$, indépendamment de l'état précédent à l'instant t. On somme donc sur tous les états précédents possibles i. Ensuite, on ajoute la vraisemblance d'émission d'observation O_{t+1}, qui ne dépend pas de l'état précédent. Donc, on a $f_j(O_{t+1})$ dehors des parenthèses. L'équation (3.30) est juste la sommation sur tous les états finaux possibles, pour obtenir la vraisemblance désirée. La procédure "arrière" s'explique de la même façon.

On voit bien, que chacun de ces deux algorithmes demande $O(L^2T)$ multiplications au lieu de $O(2TL^T)$ multiplications pour le calcul direct.

3.2.4 Solution du problème 2: Algorithme de Viterbi

Tout d'abord, en prenant le logarithme de (3.24), on définit

$$U(O,I \mid \Lambda) \triangleq \log p(O,I \mid \Lambda) = \log\left(\pi_{i_1} f_{i_1}(O_1)\right) + \sum_{t=2}^{T} \log\left(a_{i_t i_{t-1}} f_{i_t}(O_t)\right) \tag{3.34}$$

Puisque le logarithme est une fonction croissante, le problème (3.23) est équivalent au problème suivant:

$$I^* = \underset{I}{\operatorname{argmax}}\, U(O,I \mid \Lambda) \tag{3.35}$$

Ce passage au logarithme sert seulement à simplifier les calculs. Effectivement, dans l'expression (3.24) on a un produit d'un grand nombre de vraisemblances et de probabilités. Pour une valeur de T assez importante ce produit devient trop petit ou bien trop grand, provoquant des problèmes de dépassement des possibilités de représentation numérique (*underflow* ou *overflow*). Dans le domaine logarithmique ce n'est plus le cas.

Imaginons maintenant que l'on construit un graphe orienté à LT nœuds. Chaque nœud (it) représente le fait d'être dans l'état i à l'instant t en émettant l'observation O_t, et on peut aller du nœud $(i[t-1])$ au nœud (jt) avec un coût $\log(a_{ij}) + \log(f_j(O_t))$. Le coût d'un chemin dans ce graphe est la somme des coûts de tous les déplacements successifs. L'exemple d'un tel graphe pour un HMM à 3 états et $T = 4$ est représenté dans la figure 3.3. On voit bien que la solution du problème (3.35) consiste à trouver dans ce graphe le chemin avec le coût maximal. Un tel problème se résout à l'aide de la méthode de *Programmation Dynamique*. Dans le cadre des HMMs cette méthode s'appelle *l'algorithme de Viterbi*.

Notons par $\delta_t(i)$ le coût maximal accumulé à l'état i à l'instant t, c'est-à-dire le coût du meilleur chemin qui s'arrête au nœud (it), et par $\psi_t(i)$ l'état à l'instant $t-1$ qui donne le coût maximal pour la transition à l'état i à l'instant t.

Algorithme de Viterbi

1. Initialisation, pour $1 \leq i \leq L$:

$$\delta_1(i) = \log(\pi_i) + \log(f_i(O_1)) \tag{3.36}$$

$$\psi_1(i) = 0 \tag{3.37}$$

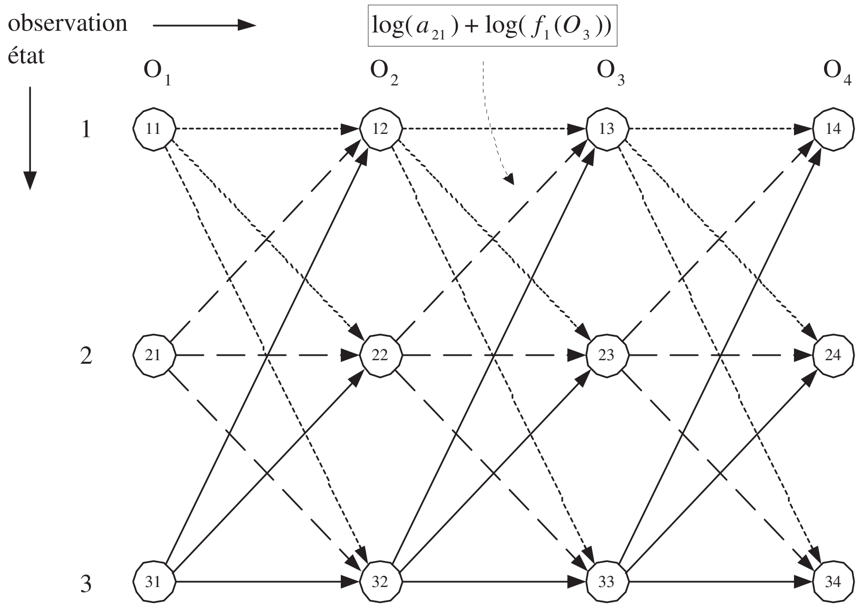

FIG. 3.3 – *Graphe pour la recherche de Viterbi (L = 3, T = 4).*

2. Calcul récursif, pour $t = 2,3,\ldots,T$, pour $1 \leq j \leq L$:

$$\delta_t(j) = \max_{1 \leq i \leq L} [\delta_{t-1}(i) + \log(a_{ij})] + \log(f_j(O_t)) \quad (3.38)$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \leq i \leq L} [\delta_{t-1}(i) + \log(a_{ij})] \quad (3.39)$$

3. Terminaison:

$$U^* = \max_{1 \leq i \leq L} [\delta_T(i)] \quad (3.40)$$

$$i_T^* = \operatorname*{argmax}_{1 \leq i \leq L} [\delta_T(i)] \quad (3.41)$$

4. Tracement en arrière de la séquence d'états optimale, pour $t = T-1, T-2, \ldots, 1$:

$$i_t^* = \psi_{t+1}(i_{t+1}^*) \quad (3.42)$$

Donc l'algorithme de Viterbi donne la séquence d'états optimale $I^* = i_1^*, i_2^*, \ldots, i_T^*$. La vraisemblance maximisant (3.24) peut être aussi calculée comme $\exp(U^*)$.

On peut facilement estimer que, comme pour la procédure "avant - arrière", la complexité de calcul est d'ordre $O(L^2T)$ au lieu de $O(2TL^T)$ pour le calcul direct.

3.2.5 Solution du problème 3: Algorithme de Baum-Welch

Le critère (3.25) qu'on veut optimiser pour résoudre le problème 3 s'appelle le critère de *Maximum de Vraisemblance (MV)*. Si nous essayons d'appliquer à ce problème une des méthodes d'optimisation classiques (celle du gradient par exemple), nous verrons très vite que cela n'est pas possible puisqu'il nous manque des données. Effectivement, nous connaissons la séquence d'observations O, mais nous ne disposons aucune connaissance sur le processus sous-jacent S sauf la distribution de chaque état S_t conditionnellement à l'état précédent (matrice A). Cette donnée non-observée s'appelle la *donnée latente*.

On a donc besoin d'une autre technique d'estimation des paramètres. *L'algorithme EM* (Expectation - Maximisation) est une solution très générale d'un tel problème. Cet algorithme est itératif. Chaque itération se décompose en deux étapes: expectation et maximisation, respectivement. Généralement il y a deux types d'applications d'algorithme EM. Le premier apparaît quand il y a une donnée non-observée, comme pour les HMMs. Le second apparaît quand la fonction de vraisemblance est assez compliquée, comme la fonction (3.16) pour les GMMs, mais qui se simplifie (3.13) quand on introduit un processus sous-jacent non-observé (voir la définition 3.2.5).

On n'en dira pas plus sur l'algorithme EM et on introduira avec des explications intuitives *l'algorithme de Baum-Welch*, qui est la version finale de l'algorithme EM pour les HMMs.

Introduisons les notations suivantes:

- $\Lambda = \{\pi, A, \mu_i, \Sigma_i \mid i = 1, \ldots, L\}$ les paramètres du modèle estimés à l'itération précédente,
- $\hat{\Lambda} = \left\{\hat{\pi}, \hat{A}, \hat{\mu}_i, \hat{\Sigma}_i \mid i = 1, \ldots, L\right\}$ les paramètres du modèle estimés à l'itération courante,
- $\gamma_t(i) = P(s_t = i \mid O, \Lambda)$ la probabilité d'être dans l'état i à l'instant t, étant donnés la séquence d'observations O et le modèle Λ,
- $\xi_t(i,j) = P(s_t = i, s_{t+1} = j \mid O, \Lambda)$ la probabilité de passer de l'état i à l'instant t à l'état j à l'instant $t+1$ sachant O et Λ.

En utilisant la définition de la probabilité conditionnelle et les expressions (3.26) et (3.27) on a:

$$\gamma_t(i) = \frac{p(s_t = i, O \mid \Lambda)}{p(O \mid \Lambda)} = \frac{\alpha_t(i)\beta_t(i)}{p(O \mid \Lambda)} \tag{3.43}$$

De la même façon on peut montrer, que $\xi_t(i,j)$ s'exprime comme:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} f_j(O_{t+1}) \beta_{t+1}(j)}{p(O \mid \Lambda)} \tag{3.44}$$

Ces probabilités sont alors exprimées à l'aide des grandeurs calculées par la procédure "avant-arrière" 3.2.3. Ensuite, si on somme $\gamma_t(i)$ et $\xi_t(i,j)$ de $t = 1$ jusqu'à $T - 1$, les quantités obtenues peuvent être considérées comme:

$$\begin{aligned} \sum_{t=1}^{T-1} \gamma_t(i) &= \text{Estimation du nombre de transitions effectuées à partir de } i \\ \sum_{t=1}^{T-1} \xi_t(i,j) &= \text{Estimation du nombre de transitions de } i \text{ vers } j \end{aligned}$$

Maintenant il est assez naturel de calculer les probabilités de transition $\hat{a}_{ij}$ du nouveau modèle $\hat{\Lambda}$ comme le rapport entre le nombre de transitions de i vers j et le nombre de transitions effectuées à partir de i. Les vecteurs moyens $\hat{\mu}_i$ et les matrices de covariance $\hat{\Sigma}_i$ sont calculées de la manière habituelle, mais en pondérant selon les probabilités $\gamma_t(i)$. On obtient alors l'algorithme suivant:

Algorithme de Baum-Welch

1. Initialisation: choisir une approximation initiale $\Lambda = \Lambda^0$,
2. Estimation des probabilités (l'étape d'expectation de l'algorithme EM): calculer $\gamma_t(i)$ et $\xi_t(i,j)$ en utilisant les expressions (3.43) et (3.44) avec la procédure "avant-arrière".
3. Réestimation des paramètres (l'étape de maximisation de l'algorithme EM):

$$\hat{\pi}_i = \gamma_1(i) \tag{3.45}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.46}$$

$$\hat{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) O_t}{\sum_{t=1}^{T} \gamma_t(i)} \tag{3.47}$$

$$\hat{\Sigma}_i = \frac{\sum_{t=1}^{T} \gamma_t(i)(O_t - \hat{\mu}_i)(O_t - \hat{\mu}_i)^T}{\sum_{t=1}^{T} \gamma_t(i)} \tag{3.48}$$

4. Poser $\Lambda = \hat{\Lambda}$ et passer à l'étape 2, ou bien arrêter selon un critère d'arrêt (par exemple un nombre d'itérations fixé).

L'algorithme EM, qui est au fond de cet algorithme, assure la convergence vers un minimum local selon l'approximation initiale Λ^0. En plus, la vraisemblance maximisée ne peut qu'augmenter à chaque itération, c'est-à-dire

$$p(O \mid \hat{\Lambda}) \geq p(O \mid \Lambda) \tag{3.49}$$

Le problème de dépassement des possibilités de représentation numérique représenté dans 3.2.4 a aussi lieu pour la procédure "avant-arrière" utilisée dans cet algorithme. Mais ce problème n'est plus évitable en passant dans le domaine logarithmique. Effectivement, à la différence de (3.24) dans (3.22) il y a une sommation, qui ne permet pas de faire cela. Une autre méthode de mise en échelle pour éviter ce problème peut être trouvée dans [1].

3.3 Modèles acoustiques

Les modèles acoustiques sont des modèles de phonèmes. Chaque phonème est modélisé par un petit HMM à cinq états de type 'gauche-droite'. L'exemple d'un tel HMM est représenté dans la figure 3.4. Ce HMM a donc 2 états non-émettant (l'état d'entrée et l'état de sortie) et 3 états émettant, qui modélisent la création, la propagation et la disparition d'un phonème. La matrice de transition n'est pas pleine et on peut soit se déplacer à droite soit rester dans l'état courant.

Il existe deux types de modèles acoustiques: *les monophones*, quand chaque phonème est modélisé indépendamment du contexte où il se trouve, et *les triphones*, quand la modélisation de chaque phonème s'effectue selon les phonèmes à gauche et à droite (voir la figure 3.5). Si on utilise un modèle HMM pour chaque triphone, sachant qu'il y a 40 phonèmes par exemple, on aura besoins de 40^3 modèles. Généralement cela n'est pas possible à cause du manque des données d'apprentissage pour un si grand nombre de modèles. Il existe donc des méthodes de regroupement des triphones selon les contextes équivalents permettant réduire le nombre de modèles. Les modèles de triphones donnent normalement une meilleure performance que ceux de monophones.

On présentera maintenant une des méthodes d'apprentissage des modèles acoustiques qu'on va utiliser. A partir d'une phrase, de sa transcription phonétique et d'un enregistrement de cette phrase prononcé par un locuteur, cette méthode consiste à effectuer les opérations suivantes:

1. A l'aide du module acoustique, obtenir une suite de vecteurs acoustiques à partir de cet enregistrement.
2. Faire une concaténation des modèles acoustiques (HMMs) selon l'ordre des phonèmes dans la transcription. On obtient donc un grand HMM, qui est aussi 'gauche-droite'.
3. Estimer les paramètres de ce grand HMM par l'algorithme de Baum-Welch prenant la suite des vecteurs acoustiques comme la séquence d'observations. On estime ainsi des paramètres des modèles acoustiques (petits HMMs).

Cette méthode est intéressante, puisqu'elle n'exige pas de connaissance des frontières des phonèmes dans l'enregistrement.

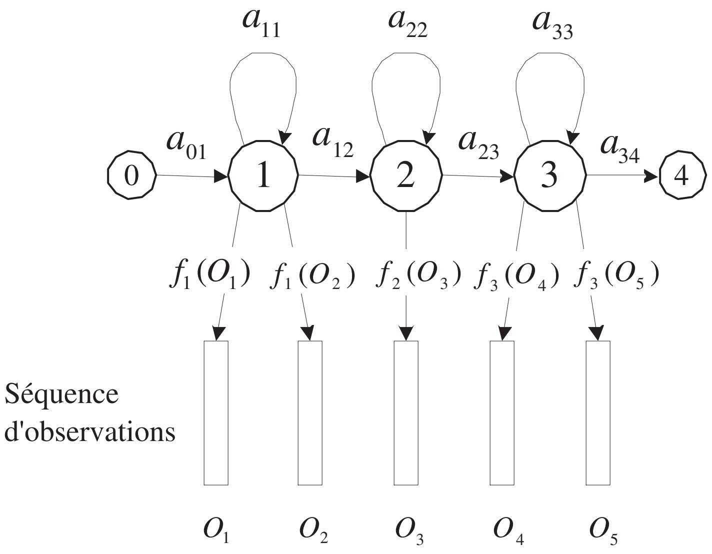

FIG. 3.4 – *HMM de type 'gauche-droite' modélisant un phonème.*

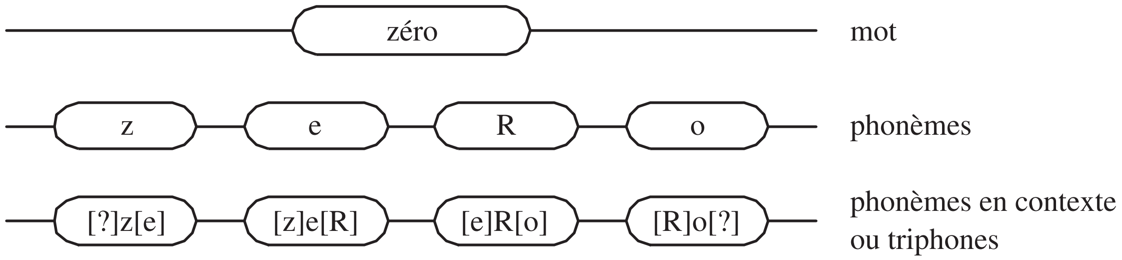

FIG. 3.5 – *Deux types de modélisation.*

3.4 Modèle de langage

Au début du développement des systèmes de reconnaissance de la parole il y avait l'idée d'introduire explicitement des règles grammaticales dans le processus de reconnaissance. Pourtant cette approche est très lourde, difficile à mettre en œuvre et elle ne rentre pas dans le formalisme commun avec les modèles acoustiques. Une autre approche, qui est basée sur une simple estimation des probabilités conditionnelles des mots sachant quelques mots précédents ne présente pas tous ces défauts. Cette approche probabiliste montre une performance similaire, ou bien meilleure, que celle utilisant des règles grammaticales.

Pour calculer $P(W)$, le score linguistique (voir la section 3.1) d'une suite de mots $W = w_1 \ldots w_i \ldots w_K$, on utilise tout d'abord la définition de la probabilité conditionnelle.

$$\begin{aligned} P(W) &= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_2 w_1) \ldots P(w_K \mid w_{K-1} \ldots w_1) \\ &= \prod_{k=1}^{K} P(w_k \mid w_{k-1} \ldots w_1) \end{aligned} \tag{3.50}$$

Ensuite, on fait une approximation en supposant que la probabilité a priori d'un mot ne dépend que des $N-1$ mot précédents, et on a:

$$P(W) \approx P(w_1)P(w_2 \mid w_1)\ldots \prod_{k=N}^{K} P(w_k \mid w_{k-1}\ldots w_{k-N+1}) \tag{3.51}$$

L'ensemble des probabilités conditionnelles $P(w_N \mid w_{N-1}\ldots w_1)$, estimées pour toutes les séquences $w_1 w_2 \ldots w_N$ de mots du vocabulaire, s'appelle ***modèle de langage N-gramme*** . Pour $N=2$ et $N=3$ les modèles de langage sont appelés ***bigramme*** et ***trigramme*** respectivement, et c'est généralement ces deux modèles qui sont utilisés en pratique.

Par exemple la probabilité a priori de la phrase "le temps est beau" sera estimée à l'aide d'un modèle trigramme comme:

$$P(\text{le temps est beau}) \approx P(\text{le})P(\text{temps} \mid \text{le})P(\text{est} \mid \text{le temps})P(\text{beau} \mid \text{temps est})$$

Les paramètres (les probabilités conditionnelles) d'un modèle de langage N-gramme sont estimés sur un grand volume de texte simplement en comptant les fréquences relatives:

$$P(w_N \mid w_{N-1}\ldots w_1) = \frac{\#(w_1 \ldots w_{N-1} w_N)}{\#(w_1 \ldots w_{N-1})} \tag{3.52}$$

où $\#(w_1 \ldots w_{N-1} w_N)$ est le nombre de fois quand la séquence $w_1 \ldots w_{N-1} w_N$ est apparue dans le texte.

Il y a aussi une difficulté liée à cette estimation. Pour n'importe quelle base de texte il existe toujours des suites de mots qui n'y apparaissent jamais. Selon (3.52) les probabilités estimées pour ces suites de mots seront égales à zéro, et donc n'importe quelle phrase contenant une de ces suites ne sera pas reconnue correctement. Pour éviter cela il existe des méthodes permettant d'associer des valeurs positives à ces probabilités [2].

3.5 Décodage

Sachant la suite de vecteurs acoustiques O, le module de décodage a pour but de retrouver la phrase $\hat{W}$ satisfaisant le critère (3.3). Puisque les modèles acoustiques modélisent les phonèmes, la vraisemblance $P(O \mid W)$ ne peut pas être calculée directement sans savoir des prononciations de W fournies par les modèles lexicaux. Supposons que la phrase W possède K prononciations possibles $\{F_k(W)\}$, cette vraisemblance peut être donc réécrite ainsi:

$$p(O \mid W) = \sum_{k=1}^{K} p(O \mid F_k(W), W) P(F_k(W) \mid W) \tag{3.53}$$

Ensuite, en supposant que les modèles lexicaux sont "simples" (les prononciations sont équiprobables, c'est-à-dire $P(F_k(W) \mid W) = 1/K$), en remarquant que la connaissance de W est déjà cachée dans la notation $F_k(W)$, et en faisant une approximation remplaçant la sommation sur toutes les prononciations possibles par un maximum, nous avons

$$p(O \mid W) = \sum_{k=1}^{K} p(O \mid F_k(W)) \approx \max_{F_k(W)} p(O \mid F_k(W)) \tag{3.54}$$

Le critère (3.3) devient alors

$$\tilde{W} = \operatorname*{argmax}_{W} \left[\max_{F(W)} p(O \mid F(W)) P(W) \right] \tag{3.55}$$

En pratique, plutôt qu'optimiser le critère (3.55) en calculant séparément $p(O \mid F(W))$ et $P(W)$ pour chaque hypothèse $(W,F(W))$, on choisit une autre approche permettant d'atteindre une complexité de calcul raisonnable. Cette approche consiste à utiliser la procédure suivante. Pour chaque prononciation d'une phrase, les modèles acoustiques (les HMMs) correspondants sont concaténés à l'intérieur de chaque mot. Ensuite les modelès de mots obtenus sont concaténés, et chaque transition entre deux mots est fournie par un poids correspondant à la probabilité conditionnelle du modèle de langage. La figure 3.6 donne l'exemple d'une telle construction pour tester les 4 hypothèses : "*elle marche*", "*elle mange*", "*il marche*" et "*il mange*". Cette construction est presque un HMM sauf que les propriétés (3.7) ne sont plus vérifiées à cause de l'introduction des probabilités conditionnelles. Néanmoins cela n'empêche pas d'utiliser l'algorithme de Viterbi pour trouver le meilleur chemin dans ce *graphe orienté valué*. La phrase contenant ce meilleur chemin est considéré comme la phrase reconnue.

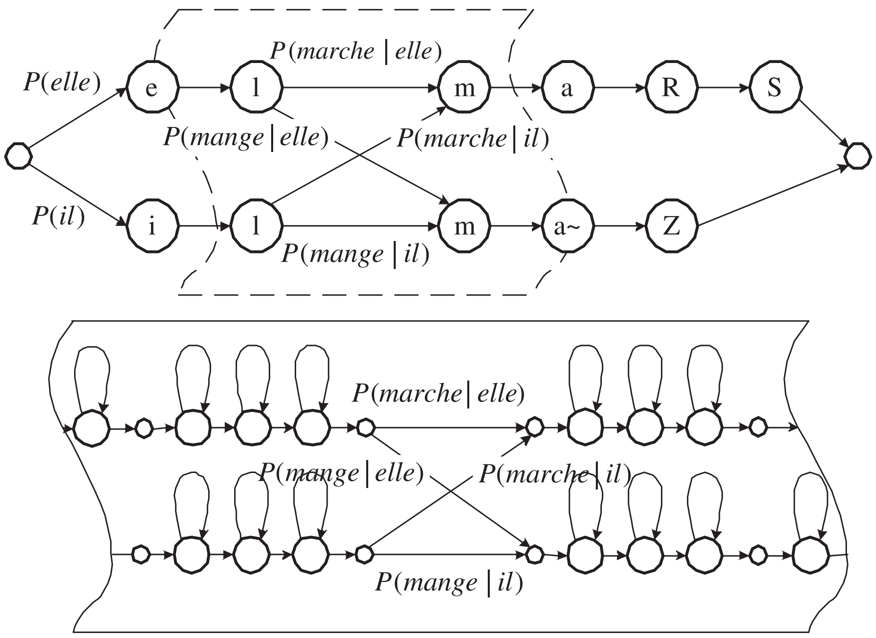

FIG. 3.6 – *Graphe orienté valué pour tester les hypothèses:* "elle marche"*,* "elle mange"*,* "il marche" *et* "il mange"*.*

Il faut remarquer que l'utilisation de l'algorithme de Viterbi introduit encore une approximation concernant le calcul de $p(O \mid F(W))$, à savoir

$$p(O \mid F(W)) = \sum_I p(O,I \mid F(W)) \approx \max_I p(O,I \mid F(W)) \tag{3.56}$$

Le vrai critère optimisé par cette procédure est donc

$$\check{W} = \operatorname*{argmax}_W \left[\max_{F(W),I_{F(W)}} p(O,I_{F(W)} \mid F(W))P(W) \right] \tag{3.57}$$

A cause des limitations de la mémoire et des capacités calculatrices il est souvent impossible en pratique de construire cet automate probabiliste pour toutes les hypothèses. De même, il est impossible de vérifier toutes les hypothèses. Cet automate est donc construit dynamiquement pendant la reconnaissance. Il existe aussi des méthodes qui permettent de ne pas vérifier toutes les hypothèses en gardant un nombre limité d'hypothèses les plus vraisemblables [4].

Chapitre 4

Analyse acoustique

Ce chapitre décrit tout d'abord le module d'analyse acoustique utilisant la représentation par des coefficients mel-cepstraux. Ensuite seront présentées des typologies de distorsions provoquant la dégradation de performance et des représentations robustes à tels types de distorsions.

4.1 Les coefficients mel-cepstraux

Comme il l'a déjà été remarqué dans la section 3.1, le but de l'analyse acoustique consiste à représenter le signal de parole sous une forme qui est plus adaptée pour la reconnaissance. Le plus souvent on utilise les représentations suivantes: MFCC (*Mel Frequency Cepstral Coefficients* en anglais) [5], LPCC (*Linear Predictive Cepstral Coefficients*) [1] ou PLP (*Perceptual Linear Predictive analysis*) [6]. Au sein de ce travail on s'intéressera surtout à la représentation MFCC qui est décrite ci-dessous.

Chaque 10 ms, une trame de la taille 20 ms est retirée du signal analysé. Soit $\{x_t(n), n = 0, \ldots, N_w - 1\}$, la trame d'analyse de numéro t, où N_w est la taille de la trame (20 ms) exprimée en nombre d'échantillons. Tout d'abord cette trame est fenêtrée par la fenêtre de Hamming et son spectre est calculé par la transformation de Fourier discrète (DFT):

$$X_t(k) = \sum_{n=0}^{N_w-1} W(n) x_t(n) e^{-\frac{2\pi i n k}{N_w}}, \qquad 0 \leq k < N_w, \tag{4.1}$$

où

$$W(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N_w - 1}\right), \qquad 0 \leq n < N_w. \tag{4.2}$$

On calcule le spectre énergétique en prenant les modules du spectre. Le spectre énergétique est ensuite moyenné par le *banc de filtres "mel"* et on obtient le *spectre de la bande critique*:

$$\mathcal{X}_t(j) = \sum_{k=0}^{K-1} \phi_j(k) |X_t(k)|, \qquad 0 \leq j < J, \tag{4.3}$$

où $K = N_w/2$. Le banc de filtres "mel" se compose de J filtres triangulaires recouvrant ϕ_j, espacés selon l'échelle "mel" correspondant aux "bandes critiques" du système auditif (voir la figure 4.1). L'échelle "mel" est définie selon la formule suivante:

$$Mel(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right), \tag{4.4}$$

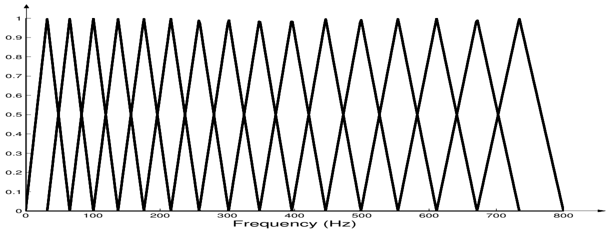

FIG. 4.1 – *Banc de filtres "mel" ($J = 16$).*

où f est la fréquence en Hz.

Ensuite on prend le logarithme du spectre de la bande critique, ce qui est aussi lié avec la sensibilité du système auditif. Puis on calcule la transformation en cosinus discrète (DCT) et on obtient les *coefficients mel-cepstraux* suivants:

$$c_m(t) = \sum_{j=0}^{J-1} V_{m,j} \log \mathcal{X}_t(j), \qquad 1 \leq m \leq J, \tag{4.5}$$

$$V_{m,j} = \sqrt{\frac{2}{J}} \cos\left(\frac{\pi m}{J}(j+0.5)\right), \qquad 1 \leq m \leq J, \quad 0 \leq j < J. \tag{4.6}$$

Généralement, on ne garde pas tous les coefficients. On ne prend par exemple que la première moitié. Pour nos expériences, nous utiliserons 24 filtres ($J = 24$) et ne garderons que les 12 premières coefficients.

Puisque les coefficients de DCT satisfont la propriété suivante

$$\sum_{j=0}^{J-1} V_{m,j} = 0, \qquad 1 \leq m \leq J, \tag{4.7}$$

les coefficients mel-cepstraux ne contiennent plus d'information sur l'énergie de signal. Effectivement, si on considère le signal $\tilde{x}_t(n) = ax_t(n)$ au lieu de $x_t(n)$, on aura:

$$\begin{aligned} \tilde{c}_m(t) = \sum_{j=0}^{J-1} V_{m,j} \log(a\mathcal{X}_t(j)) &= \sum_{j=0}^{J-1} V_{m,j} \log \mathcal{X}_t(j) + \sum_{j=0}^{J-1} V_{m,j} \log a \\ &= \sum_{j=0}^{J-1} V_{m,j} \log \mathcal{X}_t(j) = c_m(t). \end{aligned} \tag{4.8}$$

Donc les coefficients mel-cepstraux ne dépendent pas d'un facteur multiplicatif.

Pour compenser cette absence d'information énergétique, on ajoute aux 12 coefficients mel-cepstraux le logarithme de l'énergie de chaque trame:

$$E(t) = \log \sum_{n=0}^{N_w-1} W(n) x_t^2(n), \tag{4.9}$$

qui est éventuellement normalisé comme ceci:

$$\bar{E}(t) = 0.1(E(t) - E_{\max}(t)) + 1.0, \tag{4.10}$$

où $E_{\max}(t)$ est le maximum de $E(t)$ calculé sur tout le signal analysé ou sur une fenêtre.

On obtient donc le vecteur acoustique à 13 composantes $C_t = (c_1(t), \ldots, c_{12}(t), \bar{E}(t))^T$ (12 coefficients mel-cepstraux et le logarithme de l'énergie calculés pour la trame t).

Puisque la séquence de ces vecteurs acoustiques est ensuite traitée comme la séquence d'observations d'un HMM, l'information dynamique locale[1] est perdue. Effectivement, dans chaque état, toute l'information est décrite par la distribution correspondante qui ne modélise pas du tout l'ordre des données. Pour garder cette information, on étend ces vecteurs acoustiques à leurs dérivées (temporelles) premières et secondes. Ces paramètres sont souvent appelés *coefficients delta* et *delta-delta* et ils peuvent être estimés selon[2]:

$$\Delta C_t = \frac{\sum_{k=-L}^{L} k C_{t+k}}{\sum_{k=-L}^{L} k^2}. \tag{4.11}$$

La dérivée seconde $\Delta\Delta$ est calculée en itérant deux fois l'expression (4.11). On obtient finalement une suite de vecteurs acoustiques O_t, à 39 composantes chacun. Un schéma explicatif est proposé figure 4.2.

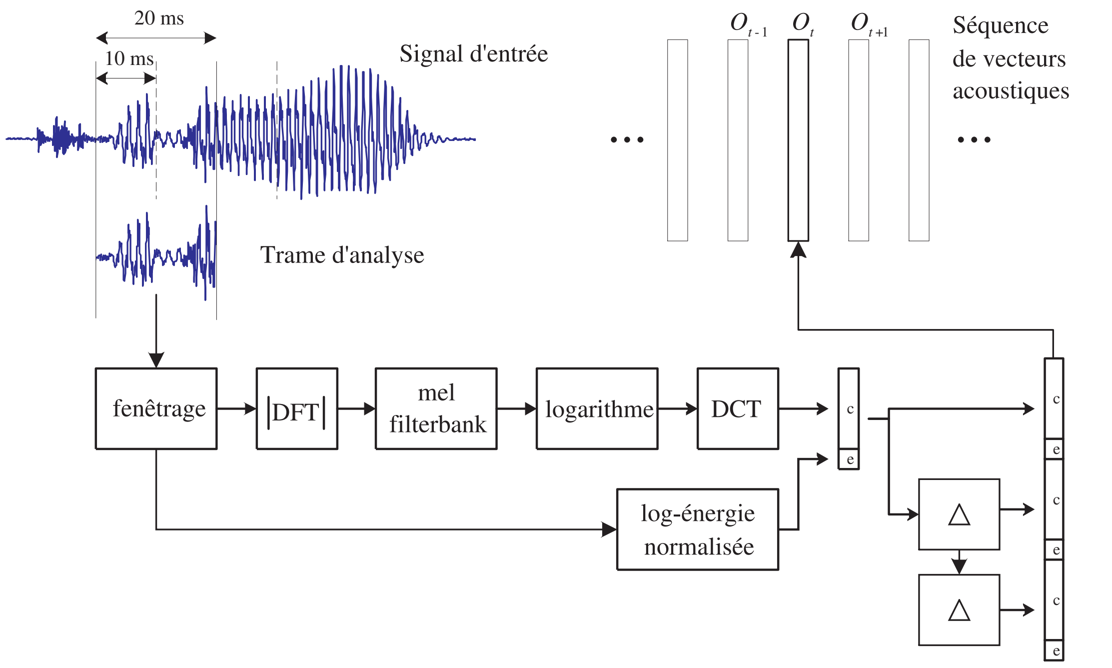

FIG. 4.2 – *Module d'analyse acoustique par la représentation MFCC.*

1. Il s'agis de la localité dans chaque état d'un HMM.
2. Cette solution est obtenue en calculant la pente de la droite minimisant un critère de moindres carrés exprimant la somme des distances entre la droite et les $2L + 1$ ponts considérés.

4.2 Typologies de distorsions

Les typologies de distorsions du signal de parole provoquant la dégradation de la performance d'un système de la reconnaissance de la parole le plus souvent considérées sont:

- *Le bruit additif* tel que le bruit de fond, celui de transmission ou celui des autres locuteurs simultanés.
- *Le bruit "convolutif"* qui émerge selon l'acoustique de la pièce ou l'équipement de l'enregistrement (le microphone, canal de la transmission).

Un tel modèle de distorsions (voir la figure 4.3) peut être décrit comme ceci:

$$y_t(n) = (x_t(n) + b_t(n)) * h(n) \tag{4.12}$$

où $x_t(n)$ est le signal de parole pur, $y_t(n)$ le signal affecté par les bruits, $b_t(n)$ le bruit additif et $h(n)$ la réponse impulsionelle du canal (bruit convolutif). t est l'indice de trame et n est l'indice à l'intérieur de chaque trame. Il n'y a pas d'indice de trame t pour $h(n)$ puisque nous supposons que le bruit convolutif varie très lentement ou même reste constant.

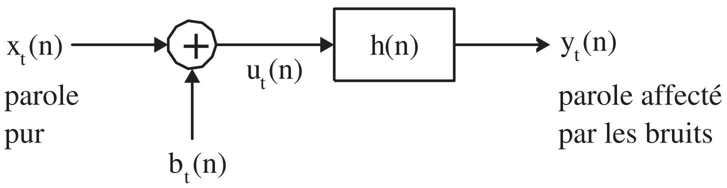

FIG. 4.3 – *Modèle de distorsions.*

4.3 Représentations robustes

Dans cette section, on introduira des méthodes permettant rendre la répresentation MFCC plus robuste aux distorsions décrites dans la section 4.2.

4.3.1 Le centrage et la réduction

Tout d'abord on introduira deux méthodes robustes aux bruits additifs et convolutifs séduisantes par leur simplicité. Ces méthodes s'appellent CMS (*Cepstral Mean Subtraction* en anglais) [7, 8, 9, 10] et VN (*Variance Normalization*) [9, 10]. Elles consistent à centrer et réduire des trajectoires cepstrales. Ces deux traitements s'appliquent aux trajectoires cepstrales avant le calcul des dérivées Δ et $\Delta\Delta$.

La CMS est la soustraction de la moyenne estimée sur une fenêtre glissante de la taille $2L+1$:

$$c'_m(t) = c_m(t) - \sum_{\tau=t-L}^{t+L} c_m(\tau) \tag{4.13}$$

Après avoir fait la CMS, on applique la VN consistant à diviser des coefficients par leur écart-type estimé sur la même fenêtre:

$$\check{c}_m(t) = c'_m(t) / \sqrt{\sum_{\tau=t-L}^{t+L} (c'_m(\tau))^2} \tag{4.14}$$

Maintenant nous expliquons pourquoi ces deux méthodes sont robustes aux bruits additifs et convolutifs. Regardons d'abord l'influence du bruit convolutif sur les coefficients mel-cepstraux. En posant $u_t(n) = x_t(n) + b_t(n)$ dans la formule (4.12) nous avons:

$$y_t(n) = u_t(n) * h(n) \tag{4.15}$$

Puisque la convolution dans le domaine temporel devient la multiplication dans le domaine spectral, $h(n)$ se transforme en facteur multiplicatif dans le domaine spectral et en facteur additif dans le domaine log-spectral. Selon (4.1) et (4.3), en gardant le même style de notations, cela s'écrit:

$$Y_t(k) = U_t(k) \cdot H(k) \tag{4.16}$$

$$\mathcal{Y}_t(j) = \mathcal{U}_t(j) \cdot \mathcal{H}(j) \tag{4.17}$$

$$\log \mathcal{Y}_t(j) = \log \mathcal{U}_t(j) + \log \mathcal{H}(j) \tag{4.18}$$

Après avoir calculé la transformation en cosinus (4.5), le bruit convolutif reste toujours dans le facteur additif puisque cette transformation est linéaire. Nous avons supposé dans la section 4.2 que le bruit convolutif varie très lentement, donc ce facteur additif est presque constant et il peut être facilement éliminé par la méthode CMS.

Il faut aussi remarquer que la méthode CMS est tout simplement le filtrage passe-haut des trajectoires cepstrales. La réponse fréquentielle d'un filtre correspondant à la CMS est représentée dans l'annexe B.

Cette technique de filtrage passe-haut dans le domaine log-spectral est souvent utilisée en traitement du signal et d'image pour enlever des distorsions convolutives. Elle est connue sous l'appellation *"égalisation aveugle"* [11] ou bien *"filtrage homomorphique"*.

Pour le cas du bruit additif, Pelecanos et Sridharan [12] donnent l'explication suivante. Supposons que le bruit convolutif est déjà éliminé avec succès, nous nous plaçons donc dans le cadre du modèle suivant:

$$y_t(n) = x_t(n) + b_t(n) \tag{4.19}$$

Dans le domaine spectral on a également:

$$Y_t(k) = X_t(k) + B_t(k) \tag{4.20}$$

Pour simplifier ce calcul nous remplaçons le module dans l'expression (4.3) par le module carré [3]. Sous cette forme la moyennisation de l'énergie dans chaque bande critique peut être considérée comme une estimation de l'espérance de l'énergie dans cette bande. On note cette estimation par $\mathbb{E}_j$ pour la bande j et on a:

3. Le module carré est aussi parfois utilisé pour le calcul de MFCC, ainsi que d'autres puissances différentes de 1 et 2.

$$\mathcal{X}_t(j) = \mathbb{E}_j(|X_t|^2) = \sum_{k=0}^{K-1} \phi_j(k)|X_t(k)|^2 \tag{4.21}$$

A partir de (4.20) on obtient:

$$\begin{aligned} \mathbb{E}_j(|Y_t|^2) = \mathbb{E}_j(|X_t + B_t|^2) &= \mathbb{E}_j(\mathrm{Re}[(X_t + B_t)^2] + \mathrm{Im}[(X_t + B_t)^2]) \\ &= \mathrm{Re}\,\mathbb{E}_j[(X_t + B_t)^2] + \mathrm{Im}\,\mathbb{E}_j[(X_t + B_t)^2] \end{aligned} \tag{4.22}$$

Supposons maintenant que les parties réelles et imaginaires des spectres de la parole X_t et du bruit B_t sont indépendantes l'une de l'autre et que le bruit est centré, c'est-à-dire $\mathbb{E}_j[B_t] = 0$. On a donc:

$$\begin{aligned} \mathbb{E}_j[(X_t + B_t)^2] &= \mathbb{E}_j[X_t^2] + \mathbb{E}_j[B_t^2] + 2\mathbb{E}_j[X_t B_t] \\ &= \mathbb{E}_j[X_t^2] + \mathbb{E}_j[B_t^2] + 2\mathbb{E}_j[X_t]\mathbb{E}_j[B_t] \\ &= \mathbb{E}_j[X_t^2] + \mathbb{E}_j[B_t^2] \end{aligned} \tag{4.23}$$

En insérant cela dans (4.22) on obtient:

$$\mathbb{E}_j(|Y_t|^2) = \mathbb{E}_j(|X_t|^2) + \mathbb{E}_j(|B_t|^2) \tag{4.24}$$

Puisque l'on suppose que l'énergie de la bande critique du bruit $\mathcal{B}_t(j)$ varie assez lentement par rapport à l'énergie de la bande critique de la parole $\mathcal{X}_t(j)$, localement, $\mathcal{B}_t(j)$ ne dépend pas de t. En prenant le logarithme de (4.24) on a localement:

$$\log(\mathcal{Y}_t(j)) = \log(\mathcal{X}_t(j) + \mathcal{B}(j)) \tag{4.25}$$

On peut en déduire que la moyenne de $\log(\mathcal{Y}_t(j))$ augmente et sa variance baisse par rapport à ceux de $\log(\mathcal{X}_t(j))$. Une explication visuelle est représentée dans la figure 4.4 à l'aide des distributions de $\mathcal{X}_t(j)$, $\mathcal{Y}_t(j)$, $\log(\mathcal{X}_t(j))$ et $\log(\mathcal{Y}_t(j))$.

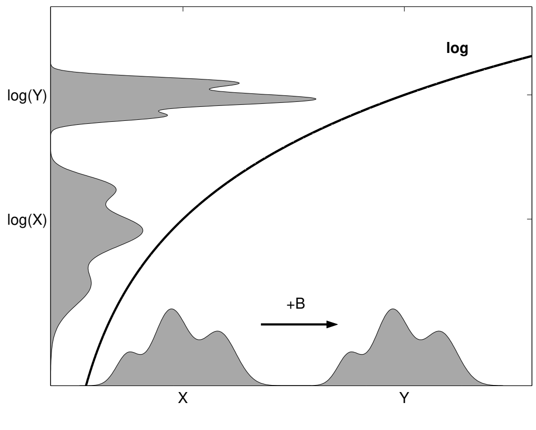

FIG. 4.4 – *Changement de la moyenne et de la variance dans le domaine log-spectral à cause du bruit additif.*

Puisque la moyenne et la variance changent dans le domaine log-spectral, ils changent aussi dans le domaine cepstral. L'utilisation des méthodes CMS et VN est donc justifiée dans le cas du bruit additif. La supposition de la constance locale de $\mathcal{B}_t(j)$ peut donner une idée sur le choix de la taille de la fenêtre glissante utilisée pour CMS et VN.

Mansour et Jaung [13] proposent une démonstration théorique montrant que le bruit blanc additif réduit surtout la norme des vecteurs cepstraux obtenus par codage à prédiction linéaire (LPCC), ce qui justifie encore une fois la réduction (VN).

Tous ces raisonnements restent valables pour la log-énergie (4.10), donc elle doit aussi être centrée et réduite.

4.3.2 Filtrage de trajectoires cepstrales

Dans cette section, nous présenterons des filtres utilisés pour le filtrage des trajectoires cepstrales. On utilise souvent des filtres passe-bande ou passe-bas.

Tout d'abord, précisons que c'est un filtrage temporel de chaque trajectoire cepstrale indépendamment des autres trajectoires et que généralement le même filtre est utilisé pour toutes les trajectoires. Ce filtrage peut s'écrire:

$$c'_m(t) = c_m(t) * h(t) \tag{4.26}$$

où $h(t)$ est la réponse impulsionelle du filtre utilisé. La log-énergie (4.10) est également filtrée. Les dérivés temporels sont calculés après le filtrage.

Il faut remarquer que, puisque le filtrage et la DCT sont des transformations linéaires et puisque le filtre est le même pour toutes les trajectoires, on peut facilement démontrer que ce filtrage dans le domaine cepstral est équivalent au même filtrage dans le domaine log-spectral. Filtrer dans le domaine cepstral est donc plus intéressent parce qu'il faut faire moins de calcul (on garde que la moitié des coefficients cepstraux). Nous utiliserons ce fait dans la suite.

Les réponses fréquentielles de tous les filtres décrits ensuite sont présentées dans l'annexe B. Puisque les trames sont retirées chaque 10 ms, le domaine fréquentiel de ces filtres s'étend jusqu'à 50 Hz. Les fréquences de ce domaine sont appelées les "*fréquences de modulation*". Ce terme, introduit par Hirsch et al. [14] pour des fluctuations temporelles des énergies sous-bande, a ensuite été aussi utilisé pour les fluctuations des coefficients cepstraux [15].

En observant (4.11), remarquons d'abord que les procédures de calcul de Δ et $\Delta\Delta$ sont aussi des filtrages passe-bande (voir annexe B).

Filtrage RASTA

Nous commencerons par le filtrage *RASTA (RelAtive SpecTrA)* introduit par Hermansky et Morgan [16]. Le filtre RASTA est un simple filtre RII passe-bande dont la fonction de transfert est

$$H(z) = 0.1\frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{1 - rz^{-1}} \tag{4.27}$$

oú le pôle $r \in (0{,}1)$ peut être réglé pour atteindre la meilleure performance du système. Selon Hermansky et Morgan [16] la valeur optimale de r est 0.94. On peut remarque que ce filtre est une combinaison de 'Δ-filtre' d'ordre $L = 2$ et d'un filtre RII du 1^{er} ordre. Hermansky applique ce filtre dans le domaine log-spectral pour la représentation PLP [6]. Selon la remarque faite au début de cette section il est raisonnable de l'appliquer dans le domaine cepstral.

Filtrage ARMA

Chen et al. [9] utilisent le filtre *ARMA (Autoregressive Moving Average)*, avec la fonction de transfert

$$H(z) = \frac{1 + z + \ldots + z^M}{2M + 1 - z^{-1} - \ldots - z^{-M}} \tag{4.28}$$

où M est l'ordre du filtre qui peut être optimisé en fonction de la performance. Ce filtre passe-bas est appliqué après avoir fait CMS et VN. Sans VN, ce schéma (CMS + ARMA) est équivalent à un filtrage passe-bande. Il est intéressant de remarquer que la réponse fréquentielle du filtre CMS + ARMA ($M = 3$) ressemble beaucoup à celle du filtre RASTA ($r = 0.94$) (voir l'annexe B).

Filtrage par des séquences de Slepian

Nadeu et al. [7] utilisent la combinaison de deux filtres RIF est utilisée.

Premièrement une étude des densités spectrales de puissance (DSPs) des trajectoires cepstrales est faite. Ces DSPs peuvent être estimées par la méthode de Welch. La méthode de Welch consiste à séparer le signal par des segments recouvrant, les fenêtrer, calculer le périodogramme de chaque segment et moyenner enfin ces périodogrammes sur tous les segments. Pour la $m^{\text{ème}}$ trajectoire cepstrale cela peut être exprimer comme:

$$D_m(\omega) = \frac{1}{K}\sum_{k=0}^{K-1}\frac{1}{F_s}\frac{\left|\sum_{\tau=0}^{T_s-1} c_m(kT_d+\tau)W(\tau)e^{-i\omega\tau T_s}\right|^2}{\sum_{\tau=0}^{T_s-1}|W(\tau)|^2} \tag{4.29}$$

où ω est la fréquence en Hz, F_s (= 100 Hz) est la fréquence d'échantillonnage, T_s est la taille de chaque segment, T_d est le décalage entre deux segments successifs, K est le nombre de segments et $W(\tau)$ est la fenêtre de Hamming (4.2) de la taille T_s. L'exemple d'une telle estimation de DSP est donné dans l'annexe C (voire la figure C.1, SNR = ∞).

Ces DSPs $D_m(\omega)$ sont moyennées sur toutes les trajectoires m et on obtient $D(\omega)$. Ensuite les auteurs remarquent que cette DSP $D(\omega)$ peut être bien approchée par la réponse fréquentielle d'une modèle *AR (Autoregressive)* du 1$^{\text{er}}$ ordre:

$$F(z) = \frac{1}{1 - rz^{-1}} \tag{4.30}$$

dont le paramètre r peut être calculé comme $r = R(1)/R(0)$, où $R(n)$ est la DFT inverse de $\{D(nF_s/T_s),\, n = 0,\ldots,T_s - 1\}$. Puisque les auteurs conseillent d'égaliser les spectres des trajectoires cepstrales dans une certaine bande, ils appliquent un filtre RIF dont la fonction de transfert est l'inverse à celle du modèle AR, c'est-à-dire

$$H(z) = 1 - rz^{-1} \tag{4.31}$$

Le deuxième filtre est un filtre passe-bas dont la largeur de bande vaut W (en Hz). La première *séquence de Slepian* (ou *discrete prolate spheroidal wave sequence*) est prise comme la réponse impulsionelle de ce filtre. La première séquence de Slepian $v_0(n)$ dépende de deux paramètres: la largeur de bande W et la longueur de cette séquence L. Elle est le vecteur propre associé à la plus grande valeur propre du système suivant:

$$\sum_{m=0}^{L-1} v_k(m)\frac{\sin(2\pi W(n-m)/F_s)}{\pi(n-m)} = \lambda_k v_k(n) \tag{4.32}$$

où $0 \le n < L$ et $0 \le k < L$. Cette séquence a la plus grande concentration d'énergie dans la bande $[0,W]$ pour une longueur L donnée. La valeur propre λ_0 vaut la fraction de l'énergie concentrée dans la bande $[0,W]$.

Malheureusement, il n'y a pas de bonnes justifications théoriques de toutes ces méthodes de filtrage. Selon Hirsch et al. [14], les fréquences de modulation existent seulement jusqu'à 25 Hz dans un signal de parole. Dans la littérature, on retrouve aussi d'autres valeurs de la fréquence supérieure de coupure possible qui varient de 12 à 25 Hz [17, 18].

Le filtrage dans des domaines autres que le domaine cepstral est aussi parfois utilisé [14, 15].

Enfin, remarquons qu'il existe des méthodes de *conception des filtres à partir des données* (*data-driven filter design* en anglais) [19, 20, 21]. Dans la section 4.3.4 nous verrons une de ces méthodes, qui est basée sur *l'analyse discriminante linéaire*.

4.3.3 Alignement de vecteurs acoustiques

Dans l'explication sur l'influence du bruit additif sur les coefficients cepstraux (section 4.3.1), on peut remarquer que le bruit additif ne change pas seulement la moyenne et la variance, mais il change aussi la forme de la distribution des coefficients cepstraux de façon non-linéaire. Cela s'explique par la non-linéarité du logarithme (voir figure 4.4) et par le fait que l'on n'a pas pris en considération tous les facteurs dans le modèle décrit dans 4.3.1. Pour avoir une idée, les histogrammes des trajectoires cepstrales de la parole affectée par le bruit additif sont représentés dans l'annexe D.

La méthode de *l'alignement de vecteurs acoustiques* (*feature warping* en anglais) [12] a pour but de résoudre ce problème. Cette méthode consiste à conditionner la distribution de chaque trajectoire cepstrale en alignant les coefficients cepstraux d'une telle façon que sa distribution soit égale à une distribution donnée. Pour un coefficient cepstral $c_m(t)$, on cherche donc $\tilde{c}_m(t)$ satisfaisant

$$\int_{-\infty}^{c_m(t)} h_m(x)dx = \int_{-\infty}^{\tilde{c}_m(t)} f(y)dy \qquad (4.33)$$

où $h_m(x)$ est la distribution de la $m^{\text{ème}}$ trajectoire cepstrale et $f(y)$ est la distribution cible. Généralement on prend comme distribution cible la distribution de la loi normale. Dans ce cas, cette méthode s'appelle aussi *gaussianisation*. Un schéma explicatif est représenté dans la figure 4.5.

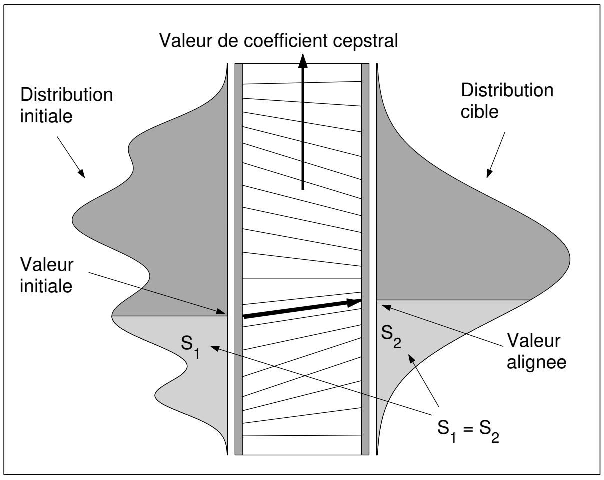

FIG. 4.5 – *Alignement de vecteurs acoustiques.*

La mise en oeuvre de cette méthode est constituée par les étapes suivantes:

1. Pour chaque coefficient $c_m(t)$ fenêtrer sa trajectoire par une fenêtre de taille L de telle façon que ce coefficient soit au milieu de la fenêtre. Il est conseillé de prendre $L = 300$ (3 sec) [12].
2. Ranger les coefficients obtenus dans l'ordre croissant de ces valeurs.
3. Calculer le rang de chaque coefficient comme sa position dans ce rangement (le plus petit coefficient obtient le rang 1 bien que le plus grande L). Cela revient à calculer la fonction de répartition empirique de la distribution locale de la trajectoire cepstrale.

4. Soit R, le rang de $c_m(t)$, calculer la nouvelle valeur $\tilde{c}_m(t)$ comme:

$$\tilde{c}_m(t) = F^{-1}\left(\frac{R-1/2}{L}\right), \tag{4.34}$$

où $F(\cdot)$ est la fonction de répartition de la distribution cible.

Certains travaux proposent d'utiliser comme distribution cible une distribution estimée à partir des données d'apprentissage [22, 23].

4.3.4 Analyse discriminante linéaire

Rappelons le principe de *l'Analyse en Composantes Principales* (*PCA* pour *Principal Component Analysis* en anglais). Considérons un ensemble de vecteurs $\{x_i\}_{1\leq i\leq N}$, $x_i \in \mathbb{R}^n$. Le but de la PCA consiste à retrouver une transformation linéaire $y = \hat{\theta}^T x$, $\hat{\theta} : \mathbb{R}^n \to \mathbb{R}^p$ avec $p < n$, orthogonale (c'est-à-dire $\hat{\theta}^T\hat{\theta} = I_p$, où I_p est la matrice unité) et telle que la somme des variances de données transformées y_i soit maximale.

Si on note par μ et Σ le vecteur moyen et la matrice de covariance des données initiales x_i:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i, \qquad \Sigma = \frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)(x_i-\mu)^T, \tag{4.35}$$

il est facile de montrer que $\theta^T\Sigma\theta$ est la matrice de covariance des données transformées. On cherche ainsi $\hat{\theta}$ comme:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \mathrm{Tr}\left(\theta^T\Sigma\theta\right), \tag{4.36}$$

où $\mathrm{Tr}(\cdot)$ symbolise la trace d'une matrice carrée. On maximise donc bien la somme des variances de y_i dans (4.36).

Si on suppose en plus que (4.36) est vérifiée pour toutes les transformations intermédiaires $\hat{\theta}_k = (\hat{v}_1, \ldots, \hat{v}_k)$, $k = 1, \ldots, p-1$ (où $\hat{\theta} = (\hat{v}_1, \ldots, \hat{v}_p)$), ce problème possède une solution unique. Cette solution consiste à prendre comme $\hat{v}_k$ le vecteur propre du problème

$$\Sigma v = \lambda v, \tag{4.37}$$

associé à la $k^{\text{ième}}$ plus grande valeur propre (par ordre décroissant) λ_k.

Il faut remarquer qu'après une telle projection les vecteurs y_i ont non seulement la plus grande variance, mais ils sont aussi décorrélés ($\hat{\theta}$ diagonalise la matrice de covariance).

Supposons maintenant que tous les vecteurs x_i sont répartis dans J classes disjointes. Notons par $l : \{1, \ldots, N\} \to \{1, \ldots, J\}$ l'application surjective déterminant l'appartenance de chaque élément à une classe, c'est-à-dire que x_i est dans la classe j si et seulement si $l(i) = j$. Admettons que nous voulons encore projeter les données sur un espace de dimension inférieure, mais cette fois-ci en gardant le plus possible de l'information discriminante entre les classes. Pour cela on peut utiliser *l'Analyse Discriminante Linéaire* (*LDA* pour *Linear Discriminant Analysis* en anglais) [24], qui est une extension de la PCA.

La LDA consiste à calculer tout d'abord les vecteurs moyens et les matrices de covariance pour chaque classe j:

$$\mu_j = \frac{1}{N_j}\sum_{i\in l^{-1}(j)} x_i, \qquad \Sigma_j = \frac{1}{N}\sum_{i\in l^{-1}(j)}(x_i-\mu_j)(x_i-\mu_j)^T, \tag{4.38}$$

où N_j est le nombre de vecteurs dans la classe j, $\sum_{j=1}^{J} N_j = N$. Ensuite on calcule les deux matrices de covariance: *inter-classe* Σ_W et *intra-classe* Σ_B (*within-class* et *between-class scatter matrices* en anglais):

$$\Sigma_W = \frac{1}{N}\sum_{j=1}^{J} N_j \Sigma_j, \qquad \Sigma_B = \frac{1}{N}\sum_{j=1}^{J} N_j(\mu_j - \mu)(\mu_j - \mu)^T, \tag{4.39}$$

Enfin on cherche la transformation $\hat{\theta}$ satisfaisant le critère suivant:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \frac{|\theta^T \Sigma_B \theta|}{|\theta^T \Sigma_W \theta|}, \tag{4.40}$$

où $|\cdot|$ désigne le déterminant d'une matrice carrée. A la différence de la trace [4], le déterminant est le produit des valeurs propres d'une matrice. C'est aussi le produit des "variances" le long de directions principales et donc c'est une mesure du carré du volume d'une hyperellipse "d'éparpillement". On voit bien que le critère (4.40) a pour but de maximiser la variabilité entre les classes tout en minimisant la variabilité à l'intérieur des classes. La solution de ce problème de maximisation consiste à prendre comme colonnes de $\hat{\theta}$ les vecteurs propres du problème à valeurs propres généralisé:

$$\Sigma_B v = \lambda \Sigma_W v, \tag{4.41}$$

associés aux p plus grandes valeurs propres. Comme pour la PCA cette solution est unique si on optimise (4.40) pour toutes les $\hat{\theta}_k$, $k = 1, \dots, p-1$, sinon ce critère est invariant aux rotations dans l'espace de la projection.

En PCA les vecteurs propres du problème (4.37) sont parfois appelés *composantes principales* (*CP*), par analogie nous allons appeler les vecteurs propres du problème (4.41) *discriminants linéaires* (*DL*). La différence principale entre la LDA et la PCA est expliquée dans la figure 4.6.

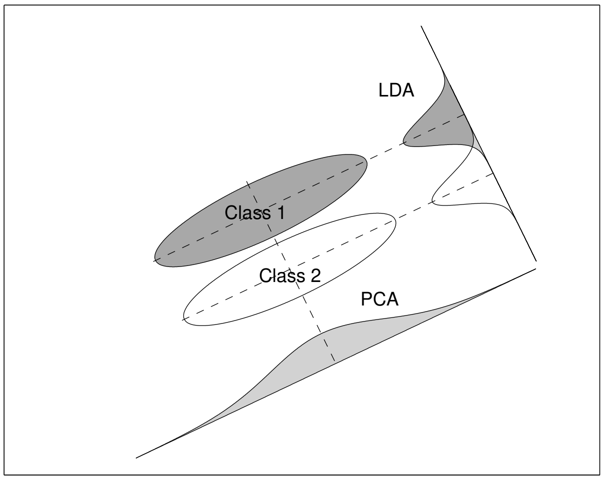

FIG. 4.6 – *La différence entre la LDA et la PCA.*

La LDA et la PCA sont beaucoup utilisées en reconnaissance automatique de la parole pour la conception des transformations linéaires dans le module d'analyse acoustique. La PCA est utilisée parce qu'elle sert à décorréler les trajectoires cepstrales [5]. La LDA augmente la discrimination entre

4. Pour PCA on peut replacer le critère (4.36) par celui maximisant $|\theta^T \Sigma \theta|$, et cela conduira au même résultat.

5. En pratique on utilise les mélanges de gaussiennes (3.16) dont les matrices de covariance sont diagonales. Ce choix est fait pour simplifier les calculs (plus besoin d'inverser la matrice) et puisque les expériences avec les matrices diagonales et le nombre de composantes de mélanges élevé donnent la même performance que avec les matrices pleines. Ce pour cette raison qu'il est intéressant d'avoir les trajectoires cepstrales décorrélées

les classes à distinguer. Ces classes peuvent être définies comme des phonèmes, des phonèmes en contexte (voir la section 3.3), des syllabes et même des états des HMMs [25, 26, 27].

Il y a aussi plusieurs façons d'appliquer ces transformations:

1. Appliquer une telle transformation à chaque trajectoire cepstrale le long de l'axe temporel. Cela revient à un filtrage et donc cette méthode s'appelle la conception des filtres à partir des données [19, 27, 28]. En utilisant cette approche on peut trouver trois filtres (les trois composantes principales de la PCA ou trois composantes discriminantes de la LDA) remplacant par exemple RASTA, RASTA + Δ et RASTA + $\Delta\Delta$. Nous verrons cette technique en détail dans la section 5.6.2.
2. Appliquer une transformation à chaque vecteur acoustique au lieu de la DCT, donc le long de l'axe fréquentiel [29]. Effectivement, après avoir fait la DCT, on ne garde que la moitié des coefficients en réduisant ainsi la dimension de l'espace. Pourquoi donc garder la DCT au lieu d'une autre transformation qui conviendrait peut être mieux?
3. La troisième approche est une unification des deux premières [30]. Elle consiste à transformer les "grands" vecteurs obtenus par la concaténation de plusieurs vecteurs acoustiques consécutifs.

Il y a aussi toutes les variations: "appliquer une telle transformation avant Δ et $\Delta\Delta$ ou bien après?", etc ...

Chapitre 5

Travail réalisé

Le but de ce chapitre consiste à détailler des aspects pratiques de ce travail. Nous commençons par la description des logiciels utilisés.

5.1 Les logiciels

Le système Sirocco consiste en plusieurs logiciels, qui sont gratuitement distribués. Ces logiciels sont:

1. Un logiciel du traitement du signal *Spro* [1]. Spro est utilisée comme module acoustique dans le système.
2. *HTK* [2] (*HMM ToolKit* en anglais) développé au CUED [3]. HTK fournit divers utilitaires pour les HMMs. Parmi ces utilitaires il y a tous les algorithmes décrits dans la section 3.2. Dans le cadre de ce travail HTK a principalement été utilisé pour l'apprentissage des modèles acoustiques.
3. Le logiciel Sirocco [4] lui-même, qui est le module de décodage du système.

5.2 La configuration du système

Maintenant précisons la configuration du système, c'est-à-dire les valeurs de certains paramètres et des méthodes utilisées pour les expérimentations.

1. *Le module d'analyse acoustique:* la représentation MFCC, chaque vecteur acoustique ayant 39 composantes.
2. *Les modèles acoustiques:* les monophones, 40 modèles (35 phonèmes et 5 modèles supplémentaires, parmi lesquels la pause courte etc.), chaque modèle est un HMM à 3 états (voir la figure 3.4), dans chaque état les observations sont modélisées par un GMM à 32 composantes, la matrice de covariance de chaque composante est diagonale.
3. *Les modèles lexicaux:* une grande liste de mots a été obtenue à partir d'une grande base de texte extrait du journal Le Monde, qui porte le même nom *Le Monde*. Un dictionnaire phonétique de cette liste a été ensuite établi à l'aide de *BDLex*, un ensemble de ressources pour la phonétisation du français.
4. *Les modèles de langage:* un modèle bigramme et un modèle trigramme également entraînés sur la base Le Monde.

---

1. http://www.irisa.fr/metiss/guig/spro
2. http://htk.eng.cam.ac.uk
3. Cambridge University Engineering Department
4. http://www.irisa.fr/sirocco/

5. *Le module de décodage:* la réduction de l'espace de recherche à l'aide d'un décodage avec le modèle bigramme suivit d'un décodage avec le modèle trigramme dans l'espace réduit. Cette technique est introduite dans [4].

5.3 Les données

Dans cette section nous présentons les données d'apprentissage et de test. Puisque le but de ce travail consiste à tester des représentations robustes aux différences entre les conditions acoustiques d'apprentissage et d'évaluation, le choix de ces données est très important.

5.3.1 Les données d'apprentissage

Le corpus *BREF* [31] consiste de 66559 enregistrements (fichiers sons) de 11002 phrases du journal Le Monde lues par 120 locuteurs. En fait ce corpus consiste en trois corpus: le corpus d'apprentissage, le corpus de développement et le corpus d'évaluation. Les nombres d'enregistrements, de phrases et de locuteurs dans chaque corpus sont résumés dans le tableau 5.1. Pour nos besoins nous appellerons par *BREF 80* et *BREF 120* le corpus d'apprentissage et le corpus BREF total respectivement. Nous allons utiliser BREF 80 comme base d'apprentissage. Les données d'apprentissage sont ainsi presque pures, excepté des bruits convolutifs (la réverbération, le microphone).

corpus	d'apprentissage	de développement	d'évaluation	total
enregistrements	41002	12913	12644	66559
phrases	3877	3624	3501	11002
locuteurs	80	20	20	120

TAB. 5.1 – *La répartition des enregistrements, des phrases et des locuteurs dans le corpus BREF.*

5.3.2 Les données de test

Comme données de test nous allons utiliser 50 phrases du corpus de la campagne AUPELF [32] qui est aussi constitué de phrases du journal Le Monde lues par plusieurs locuteurs. Pour simuler des conditions acoustiques différentes entre l'apprentissage et le test nous avons ajouté artificiellement du bruit additif à ces données. Puisque le cas du bruit additif est plus difficile à résoudre (voir section 4.3.1) il a été choisit plutôt que le cas du bruit convolutif. En plus nous allons tester deux types de bruit additif: le bruit blanc gaussien (un bruit stationnaire) et la musique (un bruit non-stationnaire). Comme musique nous avons choisis un morceau de "techno-jazz". Pour éviter le danger de l'adaptation des méthodes à un certain niveau du rapport signal-bruit (*SNR* pour *Signal to Noise Ratio* en anglais), nous avons ajouté ces bruits avec les SNRs différents (30, 25, 20, 15, 10 et 5 dB). Pour chaque configuration de la représentation MFCC nous allons donc effectuer 13 décodages (50 phrases chacun): 12 décodages pour la parole avec deux types de bruit additif et 6 SNRs et 1 décodage pour la parole pure.

5.4 Performance de la reconnaissance

Précisons la méthode d'estimation de la performance de reconnaissance. Une fois la phrase reconnue, elle va être comparée à la phrase de référence. Un alignement optimal entre ces deux phrases est effectué par la méthode de programmation dynamique. Cela nous fournit:

- H qui correspond au nombre de mots correctement reconnus,
- D qui correspond aux suppressions (deletions),
- S qui représente le nombre de substitutions,
- I le nombre d'insertions,

- N le nombre total de mots dans la vraie phrase.

La précision de reconnaissance A pour cette phrase est calculée ainsi

$$A = \frac{N - D - S - I}{N} \tag{5.1}$$

qui est éventuellement moyenné sur toutes les phrases testées. Il y a aussi une autre mesure de la performance qui est le pourcentage de phonèmes correctement reconnus ou le taux de reconnaissance. Ce pourcentage est calculé comme la précision mais sans prendre en compte le nombre d'insertions I.

5.5 Apprentissage des modèles acoustiques

L'apprentissage des modèles acoustiques est fait selon la méthode introduit dans la section 3.3. Puisque pour l'algorithme EM il y a un danger de tomber sur un minimum local qui est très loin du minimum global, l'algorithme de réestimation des paramètres est appliqué de la façon suivante:

1. Prendre les modèles avec des GMMs à une composante et les initialiser.
2. Faire 2 itérations de l'algorithme EM.
3. Arrêter si le nombre désiré de composantes de GMMs est obtenu, sinon diviser chaque gaussienne de mélanges en deux et reprendre à l'étape précédente.

Pour les GMMs à 32 composantes cela revient à faire 12 itérations de l'algorithme EM. Pour apprendre ainsi les modèles acoustiques sur le corpus BREF 80 on a besoin de 3 jours de calcul sur un PC à 2,4 GHz. Cela n'est pas très raisonnable, puisque pour tester chaque représentation il faut de nouveau apprendre les modèles. On a donc envie de partir chaque fois des modèles déjà existants. Cela n'est pas possible en utilisant l'algorithme EM tel qu'il est, puisque les modèles estimés pour une représentation sont généralement une très mauvaise approximation initiale pour estimer les modèles pour une autre représentation.

La procédure de *réapprentissage par simple passage* (*single-pass retraining* en anglais) permet de résoudre ce problème. Supposons que nous avons en disposition les *anciens modèles* correspondant à *l'ancienne représentation* et voulons construire les *nouveaux modèles* correspondant à la *nouvelle représentation*. Le réapprentissage par simple passage consiste à effectuer une étape d'expectation de l'algorithme EM (3.43), (3.44) avec les anciennes paramétrisations O poursuivie par une étape de maximisation (3.45), (3.46), (3.47), (3.48) avec les nouvelles paramétrisations O'. Après avoir fait cette procédure il vaut mieux faire quelques itérations d'algorithme EM avec les nouvelles paramétrisations pour encore améliorer les modèles. La commande **HERest** de HTK [33] fait une itération de l'algorithme EM et optionnellement elle peut effectuer le réapprentissage par simple passage.

Nous avons fait un essai de réestimation de modèles par la méthode proposée en passant de la représentation MFCC à la représentation MFCC plus le filtrage RASTA. Les évolutions des valeurs du logarithme de vraisemblance $\log(O \mid \Lambda)$ et des performances de reconnaissance au cours des itérations de l'algorithme EM sont représentées dans la figure 5.1. Selon cet essai nous avons décidé qu'il suffit de faire deux itérations supplémentaires après le réapprentissage par simple passage. Un tel calcul dure 18 heures au lieu de 3 jours.

5.6 Représentations robustes utilisées

Les méthodes de CMS, de VN, d'alignement de vecteurs acoustiques (gaussianisation) et les filtrages RASTA, ARMA, Slepian ont été codé en C au sein de Spro tels qu'ils sont représentés dans la section 4.3. Il faut juste remarquer que pour la CMS, la VN et la gaussianisation nous avons utilisé la fenêtre de la taille 300 trames (3 sec). Des combinaisons des méthodes, comme CMS + VN + ARMA par exemple, ont été aussi évaluées.

Dans les deux sections suivantes nous décrivons les filtres passe-bas que nous avons décidé d'essayer et précisons la technique de conception des filtres à partir des données.

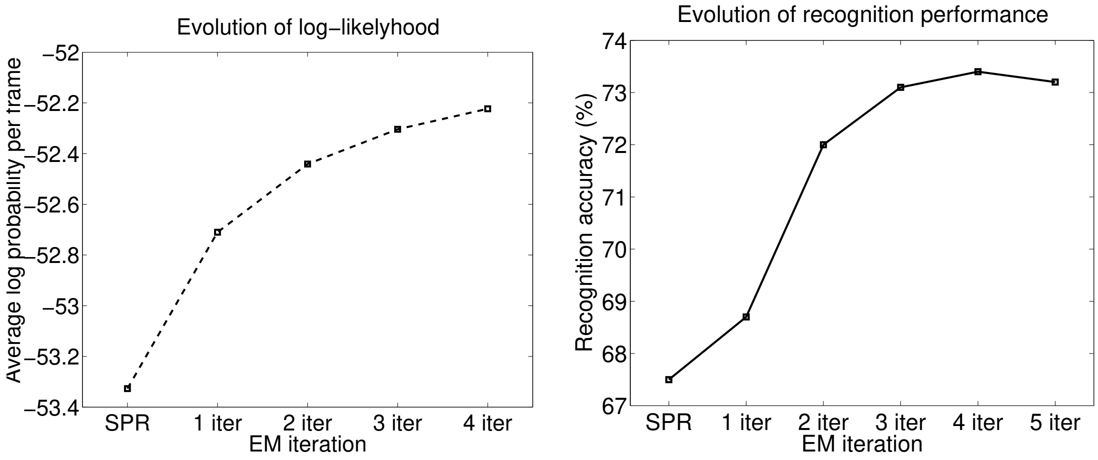

FIG. 5.1 – *Evolutions des logarithmes de vraisemblance et des performances au cours des itérations.*

5.6.1 Filtres RIF passe-bas

Tous les filtres introduits dans la section 4.3.2 sont des filtres RII, ils sont donc à phase non-linéaire. En plus les réponses fréquentielles de ces filtres ne sont pas plates dans la bande passante (voire annexe B). En espérant que des filtres passe-bas à phase linéaire ayant les réponses fréquentielles presque plates dans la bande passante seront plus avantageux, nous avons décidé de construire les filtres RIF passe-bas satisfaisant les contraintes suivantes:

- la phase linéaire,
- F_1 et F_2 $(0 < F_1 < F_2 < 50)$ sont les fréquences de coupure passe-bande et stop-bande respectivement qui vérifient $F_2 - F_1 = 3$ Hz,
- la variation de la réponse fréquentielle dans la bande passante ne dépasse pas 2 dB,
- dans la bande atténuée il y a une atténuation d'au moins 30 dB.

Comme il fut remarqué dans la section 4.3.2 la fréquence de coupure des filtres proposés dans la littérature varie entre 12 et 25 Hz. Pour comprendre où elle se trouve approximativement, nous avons choisi de construire trois filtres avec $F_1 = 12$, 19 et 25 Hz respectivement. La réponse fréquentielle de filtre avec $F_1 = 25$ Hz est représentée dans la figure 5.2.

Pour la conception de ces filtres nous avons utilisé les fonctions de Matlab **remezord** et **remez**. Pour des contraintes imposées la fonction **ordremez** calcule l'ordre minimal n, pour qu'il soit possible par la méthode de Remez de construire un filtre RIF à la phase linéaire d'ordre n satisfaisant ces contraintes. Sachant l'ordre n la fonction **remez** construit un tel filtre.

Les trois filtres ainsi construits sont d'ordre 31.

5.6.2 Conception des filtres à partir des données par LDA

Pour la conception des filtres à partir des données par LDA nous allons nous en tenir à la technique introduit par Van Vuuren et Hermansky [19] excepté quelques détails. Certains travaux [28, 30] proposent de faire la CMS avant d'appliquer la transformation LDA. Puisque celle-ci ne supprime pas toujours bien la composante fréquentielle de 0 Hz (DC). Nous avons décidé d'appliquer la CMS et la VN avant la LDA. Comme classes à discriminer, nous avons choisi 35 phonèmes. Pour pouvoir faire la LDA il faut aussi savoir à quelle classe appartient chaque vecteur acoustique, mais les signaux de BREF ne sont pas segmentés par phonèmes. Pour obtenir une telle segmentation nous allons aligner les paramétrisations des signaux avec les transcriptions phonétiques en utilisant l'algorithme de Viterbi.

La méthode réalisée se résume par les étapes suivantes:

1. Pour tous les signaux de parole de BREF 80 calculer les paramétrisations MFCC.

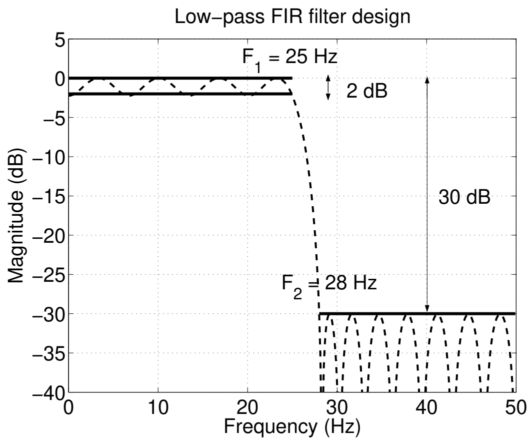

FIG. 5.2 – *Conception de filtre RIF passe-bas.*

2. A l'aide de la commande **HVite** de l'HTK aligner chaque paramétrisation avec la transcription phonétique correspondante. Plus précisément cette technique consiste à construire un HMM en concaténant les modèles acoustiques selon la transcription phonétique, puis à retrouver pour la paramétrisation (la séquence d'observations) O la meilleure séquence d'états I^* par Viterbi dans ce HMM, ensuite à faire correspondre le vecteur O_t au phonème p si l'état i_t se trouve dans le modèle acoustique ("petits" HMMs) de p.
3. Si le phonème p est aligné avec la suite des vecteurs $O_t O_{t+1} \dots O_{t+\tau}$ nous considérons que les vecteurs $O_{t+[\tau/6]}, O_{t+[\tau/6]+1}, \dots, O_{t-[\tau/6]+\tau}$ appartiennent à la classe p et le reste n'appartient à aucune classe. Cet "éloignement des bords de la segmentation" sert à diminuer l'influence de l'erreur d'alignement.
4. Ensuite calculer les paramétrisations MFCC en faisant la CMS et la VN. Les dérivées Δ, $\Delta\Delta$ ne sont pas calculées, on a ainsi des vecteurs acoustiques de dimension 13.
5. Pour chaque coefficient cepstral[5] $c_m(t)$ fenêtrer la trajectoire cepstrale par une fenêtre centrée de la taille L, c'est-à-dire extraire le vecteur

$$x_t^m = (c_m(t - [L/2]), \dots, c_m(t + L - [L/2]))^T. \tag{5.2}$$

   Nous avons choisi $L = 101$ comme il est proposé dans [19]. Si le vecteur acoustique O_t appartient à la classe p faire correspondre les vecteurs x_t^m, $m = 1, \dots, 13$ à la même classe.
6. Pour chaque trajectoire cepstrale m calculer les matrices de covariance inter et intra-classe Σ_W^m et Σ_B^m selon (4.38) et (4.39) sur toute la base BREF 80.
7. Pour chaque m trouver les trois vecteurs propres v_i^m, $i = 1,2,3$ du problème (4.41) associées aux trois plus grandes valeurs propres. Soit $v_i^m = (v_i^m(1), \dots, v_i^m(L))^T$, poser $h_i^m(t) = v_i^m(L + 1 - t)$ où $t = 0, \dots, L - 1$, $i = 1,2,3$ et $m = 1, \dots, 13$.
8. Pour obtenir la nouvelle paramétrisation de dimension 39 à partir de la paramétrisation MFCC avec la CMS et la VN il faut filtrer chaque trajectoire cepstrale m par trois filtres dont les réponses impulsionelles sont $h_i^m(t)$, $i = 1,2,3$.

Certaines caractéristiques des filtres obtenus par cette méthode sont présentées dans l'annexe E.

5. Nous considérons ici la log-énergie comme le 13$^{\text{ème}}$ coefficient cepstral

5.7 Amélioration des modèles acoustiques

Pendant ce stage un autre travail a été effectué en parallèle. Le but de ce travail consiste en l'amélioration des modèles acoustiques. Plus précisément les objectifs de ce travail sont:

1. Changement de l'alphabet phonétique de Sirocco en l'alphabet phonétique SAMPA [6] qui est un standard.
2. Changement de quelques modèles acoustiques. Notamment remplacement des modèles de phonèmes E~ et 9~ (exemples: *vin* et *brun*) par un seul modèle U~, introduction du modèle N (exemples: *camping*) qui était modélisé par deux modèles de phonèmes N et G, introduction du modèle de pause longue en plus de la pause courte.
3. Choix des meilleures transcriptions phonétiques du corpus BREF 120 parmi les transcriptions offertes par les laboratoires de l'ENST, du LIA et du LORIA. Le problème est que toutes ces transcriptions ont été obtenues automatiquement à partir du texte. Il y a donc toujours des erreurs liées par exemple aux prononciations des sigles, des noms propres etc. Ayant trois transcriptions d'une phrase on peut les aligner par Viterbi avec la prononciation correspondante et choisire la transcription la plus vraisemblable.
4. Apprentissage des nouveaux modèles acoustiques sur le corpus BREF 120 en utilisant les nouvelles transcriptions.

Ce travail a été effectué en passant par les étapes suivantes:

1. Les transcriptions de l'ENST, du LIA et du LORIA, qui avaient leurs propres alphabets phonétiques, ont été réécrites en alphabet SAMPA.
2. Les alphabets phonétiques du LIA et du LORIA ne sont pas complets, c'est-à-dire il n'y a pas de bijections entre chacun de ces alphabets et le SAMPA. Par exemple dans l'alphabet du LORIA les phonèmes Y et H (exemples: *voiture* et *puis*) ne sont pas distingués et sont représentés par un seul phonème Y. Pour remédier à cela nous avons aligné les transcriptions du LIA et du LORIA avec celles de l'ENST en utilisant la méthode de programmation dynamique. Ensuite nous avons remplacé par H les phonèmes Y du LORIA qui ont été aligné avec H de l'ENST et de même pour tous les phonèmes manquants.
3. Un graphe orienté valué a été construit pour chaque phrase. Ce graphe se compose de trois chemins qui sont chacun la concaténation des modèles acoustiques selon les transcriptions du LIA, du LORIA et de l'ENST respectivement. De plus les prononciations de liaisons et de schwas [7] ont été rendues optionnelles pour le chemin correspondant à la transcription de l'ENST. Cela a été fait à l'aide de la commande **HParse** de l'HTK. La recherche par Viterbi (commande **HVite**) dans ce graphe a été faite sur la paramétrisations MFCC du signal correspondant comme séquence d'observations. La transcription correspondant à la séquence d'états optimale a été gardée.
4. L'apprentissage des nouveaux modèles acoustiques a été fait sur le corpus BREF 120 avec les nouvelles transcriptions.

6. http://www.phon.ucl.ac.uk/home/sampa/french.htm
7. Le schwa est le phonème *e* prononcé optionnellement, comme dans *franchement* par exemple.

Chapitre 6

Les résultats expérimentaux

Les résultats d'évaluation des différentes représentations sont présentés dans les tableaux 6.1 et 6.2. Certains de ces résultats peuvent être visualisés sur la figure 6.1. Rappelons que chaque chiffre dans ces tableaux est la précision de reconnaissance évaluée sur 50 phrases du corpus de la campagne AUPELF. Les abréviations des méthodes sont assez claires. Précisons seulement que REF est la représentation de référence, c'est-à-dire la MFCC toute seule, FARP est la gaussianisation, ARMA2 est le filtrage par le filtre ARMA d'ordre 2, LPFIR19 est le filtrage par le filtre RIF passe-bas introduit dans la section 5.6.1 avec $F_1 = 19$ Hz, "+" signifie la combinaison des méthodes dont l'ordre d'implémentation coïncide avec celui représenté dans les tableaux. Les filtres de Slepian et LPFIR12 ont été aussi testés, mais ils ont donné des mauvais résultats (dégradation de la performance dans le cas de la parole pure).

Méthode \ SNR	∞	30	25	20	15	10	5
REF	75.3	74.9	73.2	67.3	55.6	27.6	13.1
RASTA	72.0	70.0	68.8	65.3	57.3	38.3	14.5
CMS	77.0	76.7	76.0	74.4	64.7	45.1	17.5
CMS + VN	77.1	77.1	75.4	72.7	68.7	55.9	30.2
FWARP	75.0	74.9	74.3	73.2	66.2	55.2	31.4
CMS + VN + LDA	68.6	69.6	67.8	64.9	59.4	44.6	24.7
ARMA2	75.9	74.9	73.3	67.4	51.1	29.8	11.4
CMS + ARMA2	76.8	76.9	76.4	73.9	64.8	43.5	17.4
CMS + VN + ARMA2	76.4	75.3	75.1	72.1	67.3	55.6	34.8
CMS + VN + LPFIR19	76.1	76.7	75.1	73.1	68.4	55.0	28.8
CMS + VN + LPFIR25	76.7	76.5	75.6	73.8	68.4	56.7	30.3

TAB. 6.1 – *Précision de reconnaissance pour la musique comme bruit additif.*

Les modèles acoustiques améliorés (voir section 5.7) ont été évalués sur 300 phrases du corpus de la campagne AUPELF au lieu de 50. Les résultats sont donnés dans le tableau 6.3.

Méthode \ SNR	∞	30	25	20	15	10	5
REF	75.3	42.9	18.9	4.8	1.2	0.5	1.9
RASTA	72.0	67.3	56.8	31.1	13.9	4.7	1.2
CMS	77.0	73.3	64.8	33.9	7.4	2.3	1.0
CMS + VN	77.1	75.8	71.4	60.5	41.2	20.1	8.4
FWARP	75.0	75.3	72.9	64.4	51.5	32.2	13.9
CMS + VN + LDA	68.6	68.9	65.9	55.7	38.2	19.8	8.0
ARMA2	75.9	39.4	20.8	3.5	0.0	0.0	1.4
CMS + ARMA2	76.8	74.1	60.9	31.4	5.7	1.2	1.5
CMS + VN + ARMA2	76.4	74.3	71.0	61.2	41.5	16.5	5.7
CMS + VN + LPFIR19	76.1	76.0	71.4	62.3	41.4	18.4	8.7
CMS + VN + LPFIR25	76.7	76.5	70.9	61.5	42.4	20.4	9.4

TAB. 6.2 – *Précision de reconnaissance pour le bruit blanc comme bruit additif.*

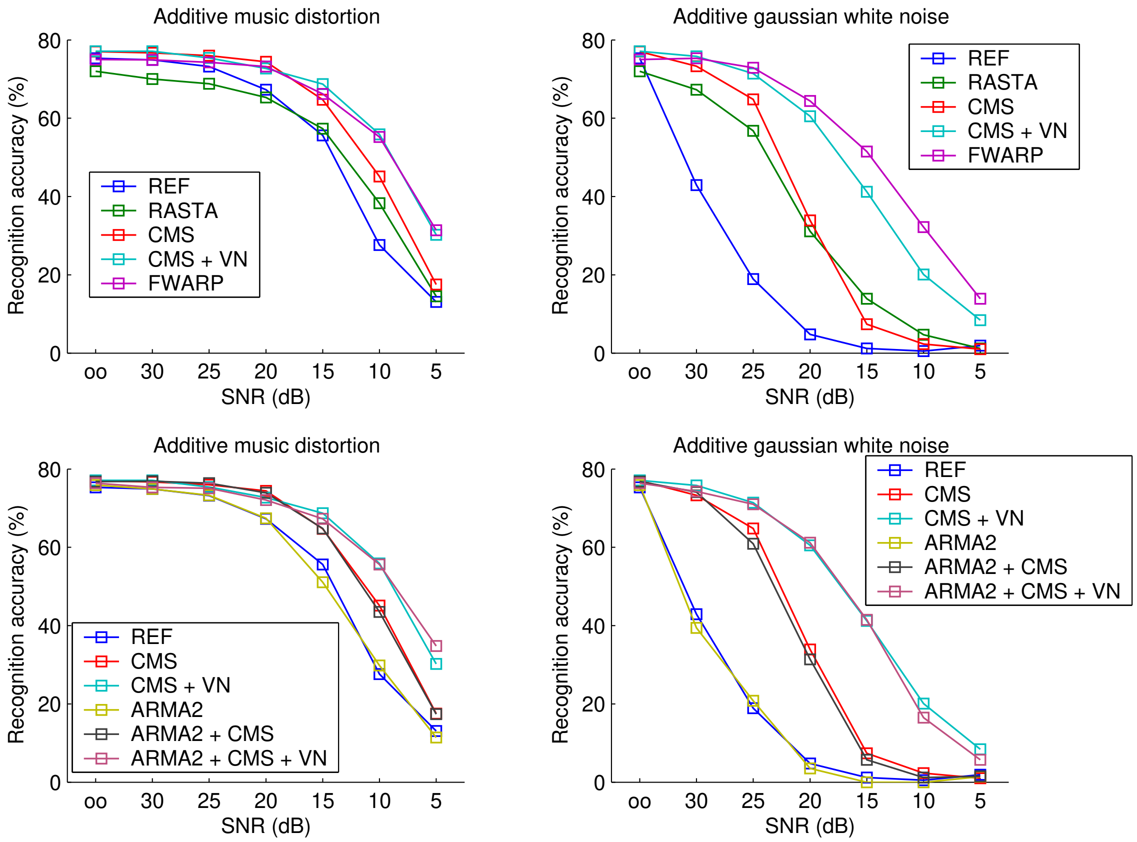

FIG. 6.1 – *Visualisation de quelques résultats.*

anciens modèles	nouveaux modèles
63.1	68.4

TAB. 6.3 – *Précision de reconnaissance avec les anciens est les nouveaux modèles.*

Chapitre 7

Discussion

Remarquons d'abord que dans le cas de la référence (REF) la performance de reconnaissance est dégradée beaucoup plus par le bruit blanc gaussien que par la musique. Cela peut s'expliquer par le fait que le spectre du bruit blanc est plat à la différence de celui de la musique. Par conséquent le bruit blanc abîme la parole dans toutes les bandes critiques, tandis que la musique ne touche que quelques bandes. En revanche, la performance des méthodes robustes est à peu près au même niveau pour ces deux types de bruit, légèrement moins bonne pour le bruit blanc.

Selon notre évaluation les méthodes CMS + VN et FWARP ont la meilleure robustesse. Ces méthodes améliorent la performance pour les deux types de bruit. Remarquons que la gaussianisation fait à la fois le centrage et la réduction, puisque après avoir fait la gaussianisation les trajectoires cepstrales ont leur moyenne à 0 et leur variance à 1. Dans le cas du bruit blanc le FWARP marche un peut mieux que la CMS + VN. En observant les histogrammes de trajectoires cepstrales (voir annexe D) nous pouvons également remarquer que la forme de la distribution de trajectoire cepstrale est modifiée beaucoup plus par le bruit blanc que par la musique. Il faut aussi remarquer que la CMS donne une petite amélioration ($\sim 2\%$) dans le cas de la parole pure. Nous supposons que c'est à cause de la différence entre les bruits convolutifs des données d'apprentissage et de test, qui est éventuellement supprimée par la CMS.

Revenons encore à l'explication théorique de l'influence du bruit additif sur les trajectoires cepstrales (voire section 4.3.1). Selon cette explication le nuage des vecteurs $\mathbb{Y}_t$ dans le domaine log-spectral est localement une déformation non-linéaire du nuage des vecteurs $\mathbb{X}_t$ le long des axes de coordonnés (voir formule (4.25)), où

$$\mathbb{Y}_t \triangleq (\log(\mathcal{Y}_t(1)), \ldots, \log(\mathcal{Y}_t(J-1)))^T, \quad \text{et} \quad \mathbb{X}_t \triangleq (\log(\mathcal{X}_t(1)), \ldots, \log(\mathcal{X}_t(J-1)))^T. \tag{7.1}$$

Pour passer dans le domaine cepstral on fait la DCT en ne prenant que la moitié des coefficients, ce qui correspond à faire tourner ces nuages dans l'espace et à les projeter sur un hyperplan. Ensuite on applique la CMS + VN ou le FWARP qui redéforme les nuages pour les rendre identiques que le signal soit bruité ou non. Mais on n'est plus dans le domaine log-spectral et cette déformation est faite le long de directions différentes par rapport à la déformation initiale. Il vaut mieux peut-être faire la CMS + VN ou le FWARP dans le domaine log-spectral. En plus Molau et al. [22] trouvent de meilleurs résultats pour la méthode de la *normalisation basée sur des histogrammes* dans le domaine log-spectral et cette méthode réalise aussi l'alignement des vecteurs acoustiques.

Le filtre passe-bande RASTA perd partiellement à la CMS et encore plus à la CMS + VN. En plus il y a une petite dégradation en cas de la parole pure. Nous pensons que ce filtre est quand même robuste par rapport à la référence parce que comme la CMS il supprime la composante fréquentielle de 0 Hz (DC). Le filtre passe-bas ARMA2 appliqué tout seul, après la CMS ou après la CMS + VN n'augmente pas la performance. Les filtres passe-bas LPFIR12, LPFIR19 et LPFIR25 appliqué après la CMS + VN n'augmente non plus la performance, mais il n'y a presque pas de dégradation dans le cas de LPFIR19 et LPFIR25. Le filtre LPFIR12 dégrade la performance. Nous pouvons donc conclure qu'il est possible de supprimer les fréquences de modulation à partir de 21 Hz (c'est la fréquence de coupure de LPFIR19, voir annexe B) sans perte en performance.

Intéressons-nous maintenant à l'étude des DSPs (voir annexe C). Puisque dans la figure C.1 les DSPs sont présentées en dB la CMS + VN correspond à bouger ces DSPs le long de l'axe d'ordonné pour qu'ils soient sur le même niveau. On voit bien qu'une telle procédure élimine bien la différence entre les trajectoires cepstrales de la parole pure et bruitée dans le domaine fréquentiel. Remarquons aussi qu'à partir de 25 Hz la DSP est plus faible qu'autour de 3 Hz d'au moins 20 dB qui correspond à un facteur 10 en échelle linéaire. Le filtrage passe-bas n'apporte donc pas grand chose au niveau de l'élimination de la différence entre les trajectoires cepstrales par rapport à la CMS + VN. C'est probablement pour cela qu'un tel filtrage n'augmente pas la robustesse. Par contre la forme de DSP dans la deuxième moitié du spectre (à partir de 25 Hz) est plus altérée par le bruit que dans la première moitié, surtout dans le cas de la musique. Dans la configuration CMS + VN + LPFIR25 une telle altération introduit une erreur dans l'estimation de la variance pour la VN, tandis que la deuxième moitié de spectre est atténuée juste après la VN. Il est peut-être plus intelligent de filtrer passe-bas les trajectoires cepstrales avant de faire le centrage et la réduction, c'est-à-dire LPFIR25 + CMS + VN par exemple.

Le filtrage par des filtres construit à partir des données par la LDA n'a pas marché. Il y a une dégradation importante de la performance même dans le cas de la parole pure. Selon [19, 20, 28] cette méthode ne marche pas moins bien que la représentation MFCC habituelle. Les caractéristiques des filtres obtenus ressemblent à ceux représentées dans [19, 20], sauf qu'il n'y a pas de suppression de la DC, mais cela est normal, parce que nous avons déjà fait la CMS. Nous pensons donc que le problème est lié avec la procédure d'apprentissage de modèles acoustiques. Soit c'est à cause de la procédure de réapprentissage par simple passage, soit à cause du petit nombre d'itération de l'algorithme EM. Malheureusement à cause du manque de temps nous n'avons pas trouvé la vraie cause. Vuuren et al. [19] comparent les filtres obtenus par la LDA avec les filtres RASTA, RASTA + Δ et RASTA + $\Delta\Delta$. Nous avons aussi fait un telle comparaison (voir figure E.1). Les réponses impulsionelles des trois discriminants linéaires ressemblent effectivement à celles de RASTA, RASTA + $\Delta\Delta$ et RASTA + Δ, les repenses fréquentielles moins. Enfin il est intéressent de remarquer que les réponses fréquentielles des deuxièmes discriminants linéaires de toutes les trajectoires cepstrales ont une atténuation forte en 3 Hz (voir figure E.2) et cette fréquence correspond à la cadence syllabique.

Chapitre 8

Conclusion et perspectives

8.1 Perspectives

A la suite de ce travail nous pouvons proposer quelques améliorations potentielles et quelques compléments, à savoir:

1. Tester deux méthodes proposées dans le chapitre précédent, à savoir le centrage et la réduction ou la gaussianisation dans le domaine log-spectral et le filtrage passe-bas avant le centrage et la réduction.
2. Faire une expérimentation sur le réglage de la taille de la fenêtre glissante utilisée pour la CMS, la VN et le FWARP.
3. Comprendre pourquoi les filtres construit par la LDA n'ont pas marché.
4. Evaluer ces représentations sur des autres types de bruits additifs (bruits de voiture, d'office etc.). Ajouter en plus les bruits convolutifs. Enfin tester sur les signaux bruités dans les conditions réelles.
5. Utiliser les données d'apprentissage qui sont aussi affectées par des bruits différents des bruits des données de test.
6. Tester ces représentations sur une autre tâche telle que la reconnaissance automatique du locuteur (RAL). A priori elles ne doivent pas marcher de la même façon, puisque l'information discriminante à garder dans la représentation cepstrale en RAL n'est pas la même qu'en RAP.

8.2 Conclusion

Une étude comparative des représentations robustes aux différences entre les conditions acoustiques d'apprentissage et d'évaluation en reconnaissance automatique de la parole a été faite. Les approches les plus simples au niveau de l'implantation ont montré les meilleurs résultats. Il y a de plus des justifications théoriques pour ces approches. Quelques idées sont apparues pendant le travail de rédaction de ce rapport et n'ont donc pas pu être mise en oeuvre.

Bibliographie

[1] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition.* Englewood Cliffs, N.J.: Prentice Hall, 1993.

[2] René Boite, Hervée Bourlard, Thierry Dutoit, Joel Hancq, and Henri Leich. *Traitement de la parole.* Presses polytechniques et universitaires romandes, CH - 1015 Laussanne, 2000.

[3] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi. An introduction to the application of the theory of probabilistic functions of Markov process to automatic speech recognition. *The Bell System Technical Journal*, 62(4):1035–1074, april 1983.

[4] Stefan Ortmanns and Hermann Ney. A word graph algorithm for large vocabulary continuous speech recognition. *Computer Speech and Language*, pages 43–72, 1997.

[5] Rivarol Vergin, Douglas O'Shaughnessy, and Azarshid Farhat. Generalized mel frequency cepstral coefficients for large-vocabulary speaker-independent continuous-speech recognition. *IEEE Transactions on Speech and Audio Processing*, 7(5), september 1999.

[6] Hynek Hermansky. Perceptual linear predictive (PLP) analysis for speech. *J. Acoust. Soc. Am.*, pages 1738–1752, april 1990.

[7] Climent Nadeu, Pau Pachès-Leal, and Biing-Hwang Juang. Filtering time sequences of spectral parameters for speech recognition. *Speech Communication*, pages 315–332, 1997.

[8] Kuo-Hwei Yuo and Hsiao-Chuan Wang. Robust features derived from temporal trajectory filtering for speech recognition under the corruption of additive and convolutional noises. In *ICASSP*, page 577, 1998.

[9] Chia-Ping Chen, Jeff Bilmes, and Kartin Kirchhoff. Low-resource noise-robust feature post-processing on Aurora 2.0. In *ICSLP*, pages 2445–2448, 2002.

[10] Sirko Molau, Michael Pitz, Ralf Schluter, and Hermann Ney. Computing mel-frequency cepstral coefficients on the power spectrum. In *ICASSP*, may 2001.

[11] Jr. Thomas G. Stockham, Thomas M. Cannon, and Robert B. Ingebretsen. Blind deconvolution through digital signal processing. In *Proceedings of the IEEE*, volume 63, pages 678–692, april 1975.

[12] Jason Pelecanos and Sridha Sridharan. Feature warping for robust speaker verification. In *Proc. ISCA Workshop on Speaker Recognition - 2001: A Speaker Oddyssey*, june 2001.

[13] David Mansour and Biing Hwang Jaung. A family of distortion measures based upon projection operation for robust speech recognition. *IEEE transactions on acoustics, speech, and signal processing*, 37(11):1659–1671, november 1989.

[14] H. G. Hirsch, P. Meyer, and H. Ruehl. Improved speech recognition using high-pass filtering of subband envelopes. In *EUROSPEECH*, pages 413–416, Genova, 1991.

[15] Brain A. Hanson and Ted H. Applebaum. Subband or cepstral domain filtering for recognition of lombard and channel-distorted speech. In *ICASSP*, pages 79–82, 1993.

[16] Hynek Hermansky and Nelson Morgan. RASTA processing of speech. *IEEE transaction on speech and audio processing*, 2(4):587–589, october 1994.

[17] Hynek Hermansky and Pratibha Jain. Down-sampling speech representation in ASR. In *EUROSPEECH*, Budapest, 1999.

[18] Noboru Kanedera, Hynek Hermansky, and Takayuki Arai. On properties of modulation spectrum for robust automatic speech recognition. In *Proc. IEEE international conferencce on acoustics, speech and signal processing*, volume 2, pages 613–616, 1998.

[19] Sarel van Vuuren and Hynek Hermansky. Data-driven design of RASTA-like filters. In *EUROSPEECH*, pages 409–412, Rhodes, Greece, 1997.

[20] Carlos Avendano, Sarel van Vuuren, and Hynek Hermansky. Data based filter design of RASTA-like channel normalisation in ASR. In *ICSLP*, pages 2087–2090, Philadelphia, october 1996.

[21] Hynek Hermansky, Eric A. Wan, and Carlos Avendano. Speech enhancement based on temporal processing. In *IEEE ICASSP*, pages 405–408, Detroit, 1995.

[22] Sirko Molau, Michael Pitz, and Hermann Ney. Histogram based normalisation in the acoustic feature space. In *Proc. of ASRU*, Italy, 2001.

[23] Florian Hilger and Hermann Ney. Quantile based histogram equalization for noise robust speech recognition. In *EUROSPEECH*, pages 1135–1138, 2001.

[24] R. O. Duda and P. B. Hart. *Pattern classification and scene analysis.* Wiley, New York, 1973.

[25] Jacques Duchateau, Kris Demuynck, Dirk Van Compernolle, and Patrick Wambacq. Class definition in discriminant feature analysis. In *EUROSPEECH*, pages 1621–1624, Scandinavia, 2001.

[26] K. Beulen, L. Welling, and H. Ney. Experiments with linear feature extraction in speech recognition. In *Proc. Europ. Conf. on Speech Communication and Technology*, pages 1415–1418, Madrid, Spain, 1995.

[27] Michael L. Shire. Data-driven modulation filter design under adverse acoustic conditions and using phonetic and syllabic units. In *EUROSPEECH*, september 1999.

[28] Jeih weih Hung, Hsin min Wang, and Lin shan Lee. Comparative analysis for data-driven temporal filters obtained via Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) in speech recognition. In *EUROSPEECH*, 2001.

[29] Hynek Hermansky and Narendranath Malayath. Spectral basis functions from discriminant analysis. In *ICSLP*, Sydney, Australia, november 1998.

[30] Sachin S. Kajarekar, B. Yegnanarayana, and Hynek Hermansky. A study of two dimensional linear discriminant for ASR. In *ICASSP*, volume 1, 2001.

[31] Lori F. Lamel, J.-L. Gauvain, and M. Eskénazi. BREF, a large vocabulary spoken corpus for French. In *EUROSPEECH*, pages 505–508, 1991.

[32] Jean-Marc Dolmazon, Frédéric Bimbot, Gilles Adda, Marc El Bèze, J. C. Caërou, Jérôme Zeilinger, and Martine Adda-Decker. Organisation de la première campagne AUPELF pour l'évaluation des systèmes de dictée vocale. In *Actes des Journées Scientifiques et Techniques du Réseau Francophone d'Ingéniérie de la Langue de l'AUPELF-UREF*, pages 13–18, 1997.

[33] Steve Young, Dave Ollason, Valtcho Valtchev, Dan Kershaw, Julian Odell, and Phil Woodland. *The HTK book.* Entropic Ltd, 1999.

Annexe A

Quelques explications sur la notion de vraisemblance

En faisant le passage entre les HMMs à lois discrètes et les HMMs à densités continues (voir définitions 3.2.3 et 3.2.4), on a remplacé la notion de probabilité d'observation par la notion de vraisemblance d'observation. Puisqu'il nous reste encore des lois discrètes, celles décrivant le modèle de base, nous avons besoin de pouvoir décrire les lois conjointes et conditionnelles des variables aléatoires discrètes et continues. Donc nous avons presque envie de travailler avec cette notion de vraisemblance comme si c'était la probabilité. Cela est possible dans une certaine manière.

Considérons une variable aléatoire discrète s à valeurs dans $\{1,2,\dots,J\}$ et une variable aléatoire continue scalaire o (O est une réalisation de cette variable). Ensuite on introduit les notions suivantes:

- $p(O,s=j) = \lim\limits_{h\to 0} \frac{P(O-h<o\leq O+h,s=j)}{2h}$ la vraisemblance conjointe des variables aléatoires o et s.
- $p(O \mid s=j) = \lim\limits_{h\to 0} \frac{P(O-h<o\leq O+h \mid s=j)}{2h}$ la vraisemblance conditionnelle de o, sachant l'événement $\{s=j\}$.
- $P(s=j \mid O) = \frac{p(O,s=j)}{p(O)}$ la probabilité conditionnelle de l'événement $\{s=j\}$, sachant que la variable aléatoire o a pris la valeur O.

On peut également écrire la règle de Bayes pour les événements $\{s=j\}$ et $\{O-h<o\leq O+h\}$. En divisant ensuite par $2h$ et en passant à la limite, on obtient la règle de Bayes sous la forme (3.2). Donc toutes les lois, qui seront utilisées, restent justifiées et il est possible de travailler avec cette vraisemblance comme avec la probabilité. Il ne faut quand même pas oublier que ce n'est pas la probabilité. Par exemple, la vraisemblance peut être plus grande que 1.

Annexe B

Les réponses fréquentielles de filtres utilisés

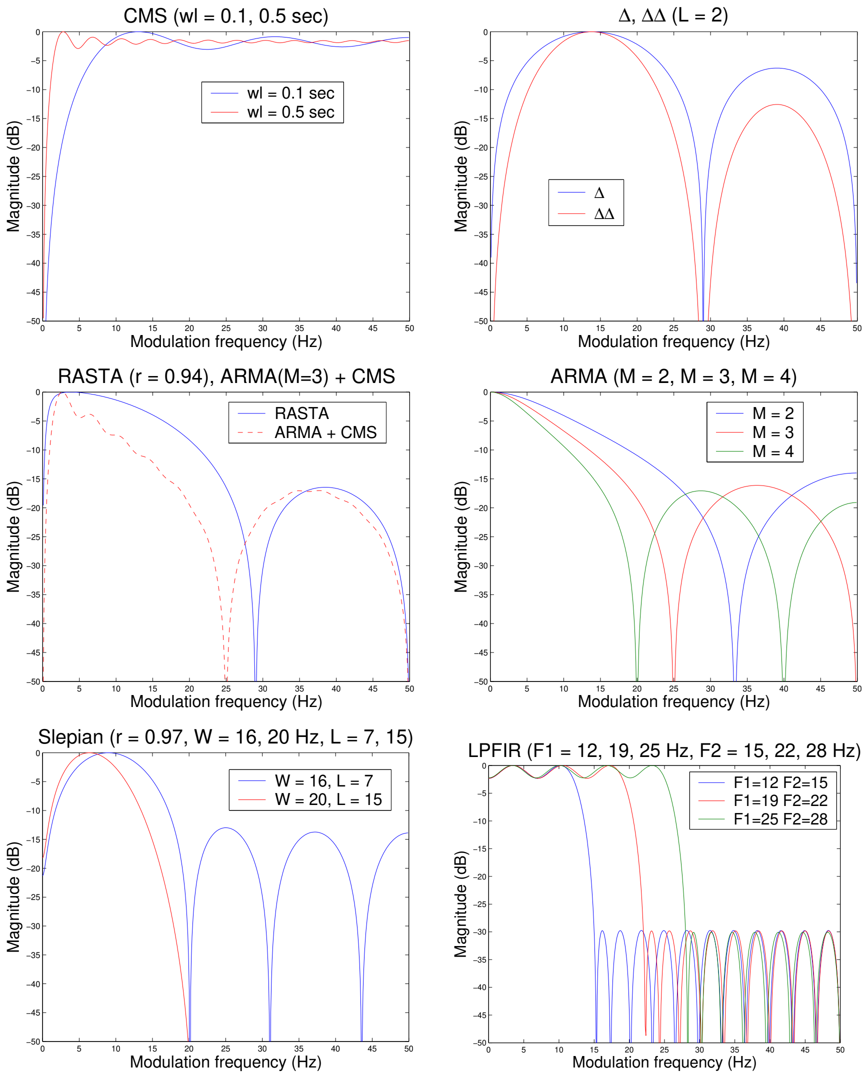

Fig. B.1 – *Les réponses fréquentielles de filtres utilisés.*

Annexe C

Etude des DSPs

Pour mieux comprendre l'influence du bruit additif sur les trajectoires cepstrales de la parole dans le domaine fréquentiel, nous présentons ici des estimations de densités spectrales de puissance (DSPs) des trajectoires cepstrales de la parole affectée par un bruit additif. Ces estimations sont faites par la méthode de Welch (4.29) pour deux types de bruit additif: la musique et le bruit blanc gaussien. Dans la figure (C.1) on représente les DSPs pour la parole pure (SNR $= \infty$), affectée par le bruit additif (SNR $= 15$, 5 dB) et pour le bruit lui-même.

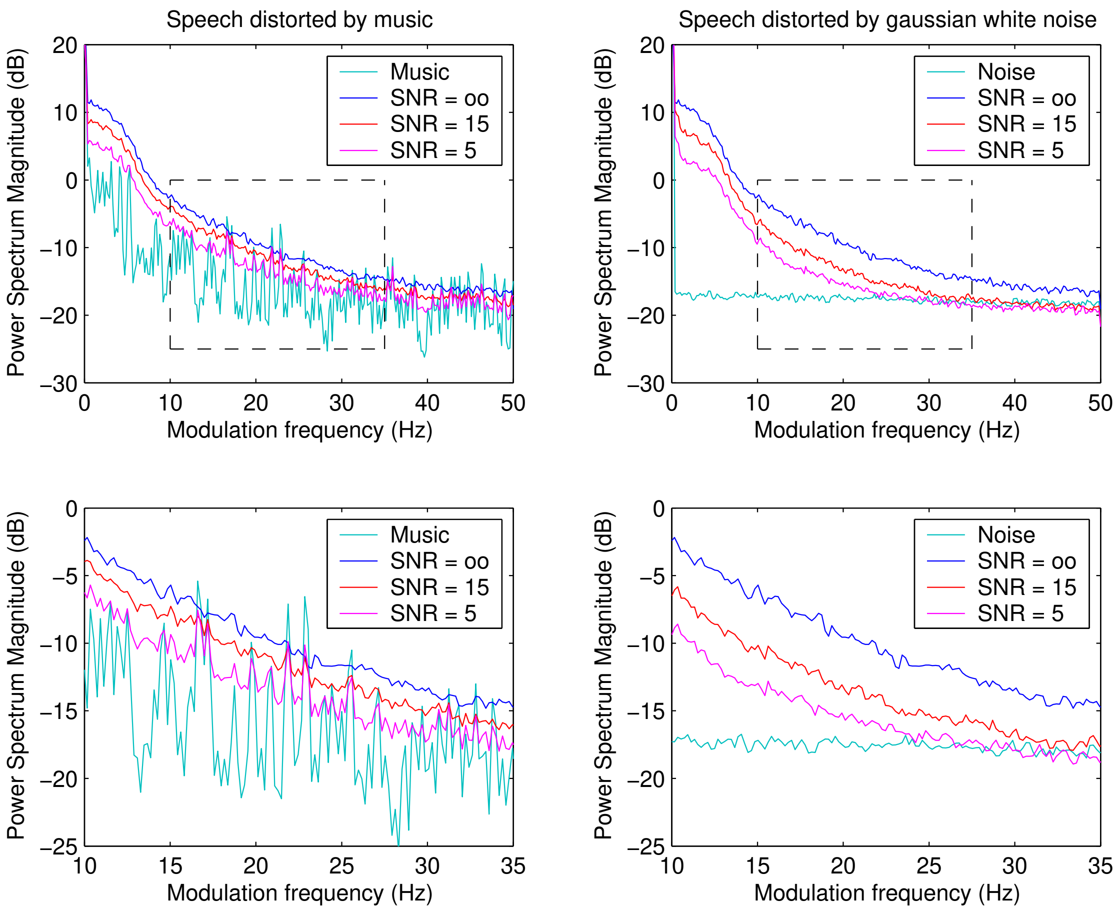

FIG. C.1 – *Estimations de densités spectrales de puissance de la* $1^{ère}$ *trajectoire celstrale.*

Annexe D

Histogrammes de trajectoires cepstrales

Les histogrammes des $1^{\text{ères}}$ trajectoires cepstrales de la parole, de la parole plus musique et de la parole plus le bruit blanc gaussien (SNR = 10 dB) sont représentés en haut de la figure D.1. Les trajectoires elles-mêmes sont représentées en bas de cette figure.

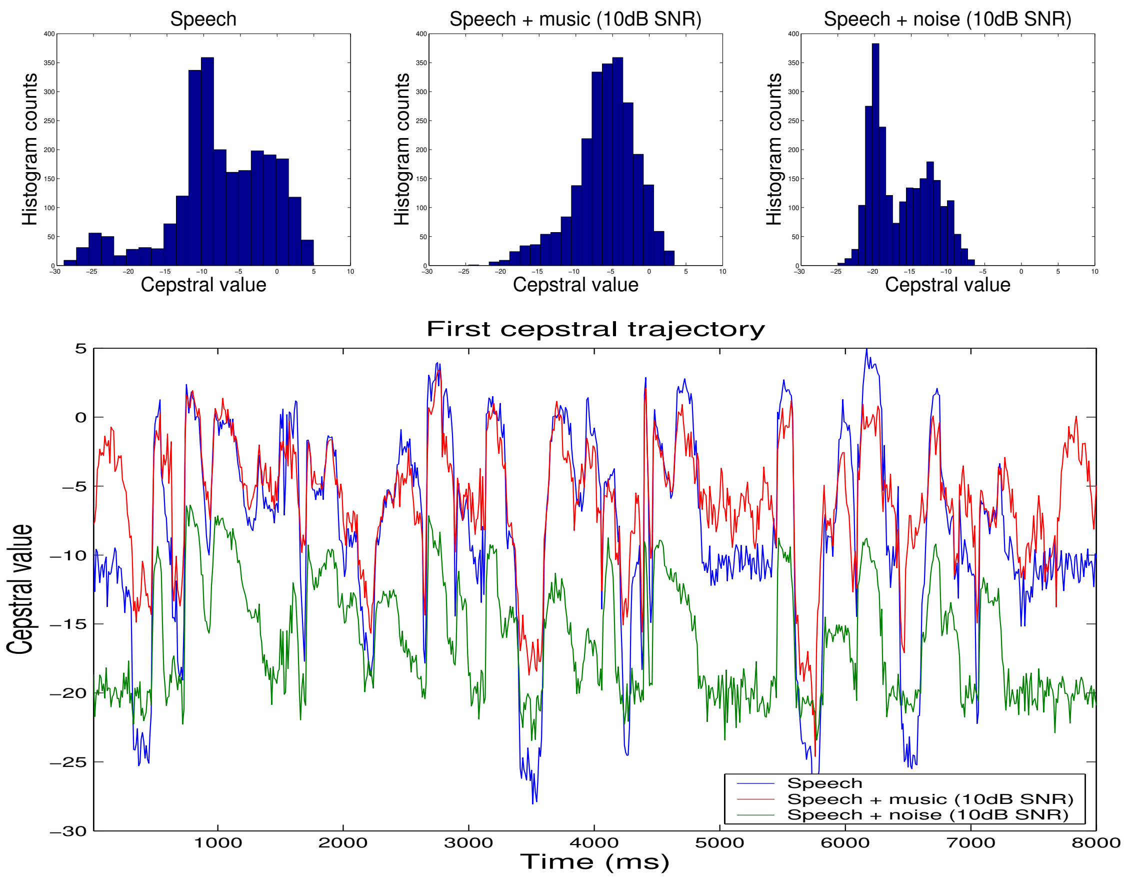

FIG. D.1 – *Les trajectoires cepstrales et ces histogrammes.*

Annexe E

Les résultats de LDA

Dans la figure E.1 nous avons représenté les réponses fréquentielles et impulsionelles de trois discriminants linéaires (*LD* pour *linear discrimiant* en anglais) obtenues par la LDA pour la $1^{\text{ère}}$ trajectoire cepstrale. Les réponses fréquentielles et impulsionelles des filtres RASTA, RASTA + $\Delta\Delta$ et RASTA + Δ sont aussi représentées pour pouvoir les comparer avec ceux de la LDA. En plus les valeurs propres correspondant à ces trois discriminantes et les matrices intra et inter-classe sont données en bas de cette figure.

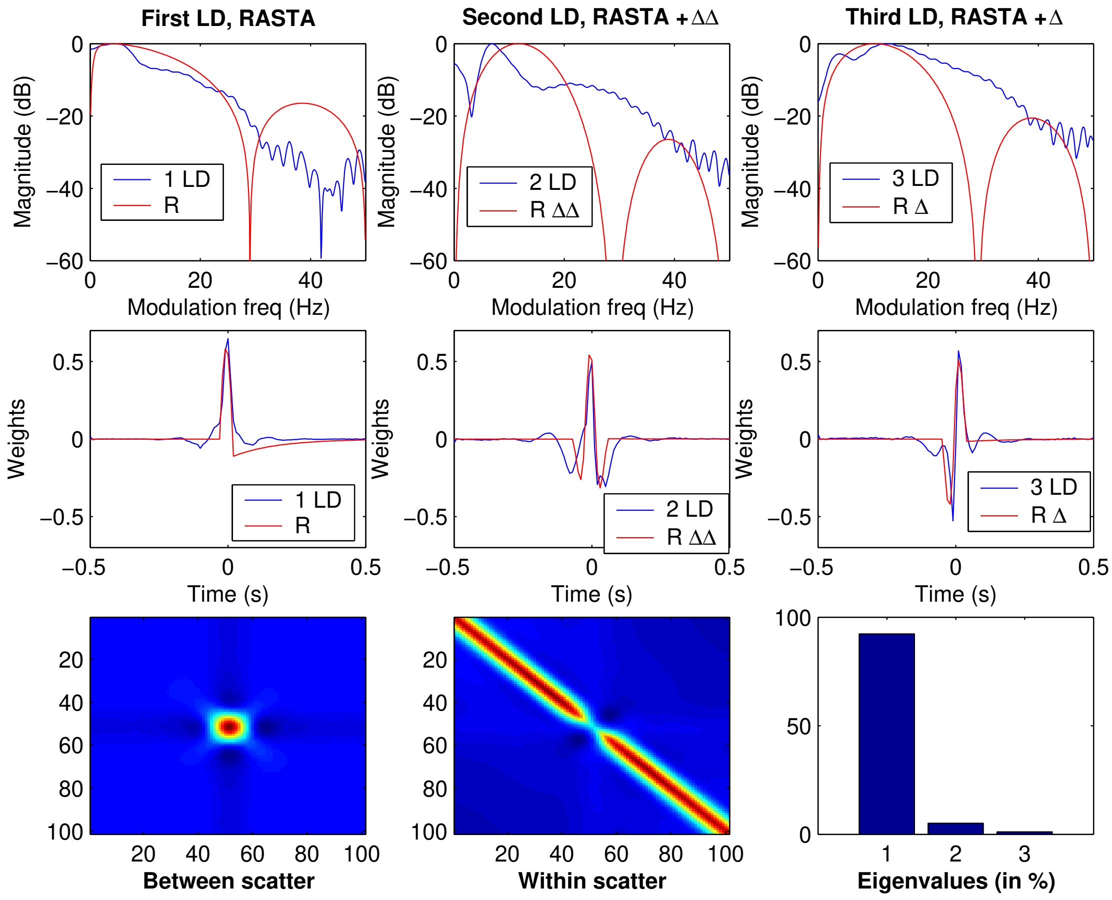

FIG. E.1 – *LD1, LD2 et LD3 contre RASTA, RASTA +* $\Delta\Delta$ *et RASTA +* Δ.

Les réponses fréquentielles des filtres obtenus par la LDA pour toutes les trajectoires cepstrales sont représentées dans la figure E.2.

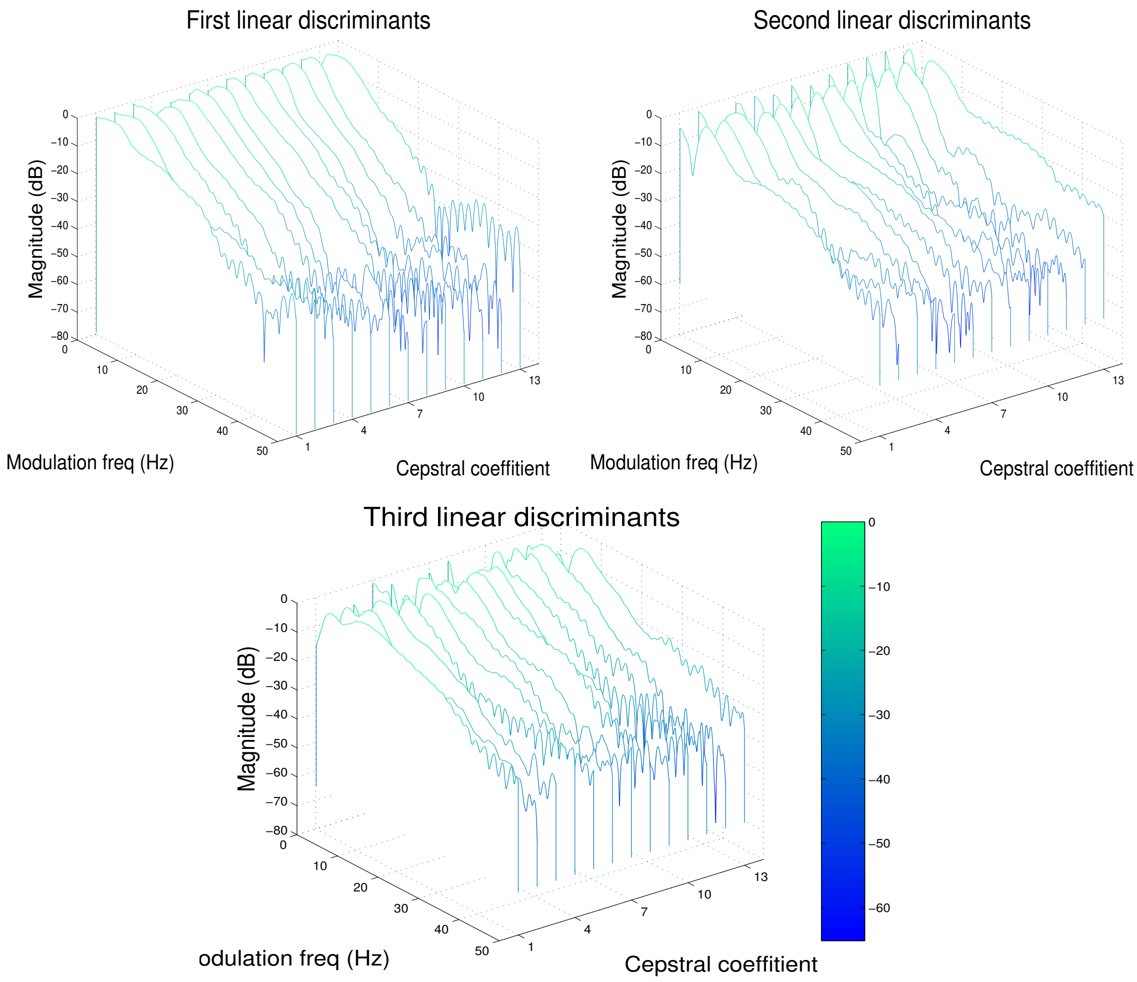

FIG. E.2 – *Les réponses fréquentielles des filtres obtenus par la LDA.*