Modélisation et reconnaissance d'événements musicaux en temps réel

Rapport d'avancement

IRCAM - CNRS - Université Paris 6

Equipes Analyse/Synthèse, Interactions Musicales Temps Réel

Sous la direction de Xavier Rodet et Norbert Schnell

Julien Bloit

Mars 2007

Résumé

Ce travail concerne la segmentation et la reconnaissance d'événements musicaux en temps réel, dans un flux audio monophonique (un seul instrument, non-polyphonique). Le contexte applicatif visé est celui de l'interaction musicien-machine (un interprète et une partition électronique). Un travail bibliographique et la définition d'un scénario applicatif visé ont permi de préciser les problématiques de la thèse.

En nous inspirant de la littérature fournie qui existe à propos de la modélisation d'unités acoustiques par des modèles de Markov cachés (en reconnaissance de parole, en transcription musicale ou en suivi de partition), nous avons mené des études préliminaires nous permettant d'expérimenter des méthodes adaptées aux problématiques identifiées : la modélisation sans expert (sélection de modèles à partir des données d'apprentissage), l'apprentissage sur peu de données (qui impose l'utilisation de modèles parcimonieux) et le décodage dans un flux.

Pour les deux premiers axes, nous avons mis en œuvre une méthode de sélection de modèle par un critère BIC (*Bayesian information criterion*) sur des données synthétiques. Les résultats préliminaires semblent indiquer que pour une tâche de classification donnée, notre méthode permet de sélectionner des modèles moins complexes que les modèles génératifs responsable des données. Nous avons également pu observer de bonnes performances en classification. Pour la reconnaissance dans un flux, nous avons mis en place une méthode de décodage Viterbi à court terme, en nous inspirant de l'algorithme *Short-Time Dynamic Time Warping*. Cette méthode doit être évaluée, notamment en ce qui concerne le temps de latence du décodage. Nous présentons les perspectives de travail à venir soulevées par nos résultats préliminaires.

Table des matières

0.1 Enjeux applicatifs

0.1.1 Contexte

Notre travail s'inscrit dans le cadre plus général de la transcription musicale qui consiste à extraire automatiquement de l'information musicale d'un signal audio émis par une ou plusieurs sources instrumentales. On peut distinguer un large éventail de problématiques selon les applications et le contexte musical visés
Certaines s'appliquent à reconnaître quels instruments sont joués dans un enregistrement polyphonique [Ess05], d'autres à repérer aussi précisément que possible tous les débuts de notes [BDA+05], à séparer les différentes sources instrumentales [Vin04], ou à extraire extraire la mélodie [RK05a] et le tempo [She98].

Cette extraction d'informations symboliques à partir du signal peut advenir dans une situation musicale interactive, comme l'accompagnement automatique d'un interprète par un traitement audio ou, un processus de synthèse, voire par une improvisation [ABC+06]. Si dans cette situation le musicien interprète une partition, l'accompagnement peut baser son évolution sur une détection de la progression de l'interprète dans la partition. On parle de *suivi de partition* pour désigner cette détection [Rap99], [SCS05]. Ce type de dispositif travaille nécessairement en temps-réel, c'est-à-dire avec une durée bornée entre l'instant où des données audio parviennent au système, et celui auquel une information est renvoyée. Les données audio disponibles au moment du traitement sont évidemment tronquées entre l'instant courant et tous les suivants, ce qui leur confère la qualification de *flux* audio, et oblige les algorithmes à recourir à des méthodes causales.

0.1.2 Système visé

En l'état actuel, la majorité des algorithmes de transcription, et le suivi de partition en particulier, considèrent *a priori* que l'information musicale est structurée en une séquence de silences et de notes, c'est-à-dire en segments qui ont des valeurs de hauteur, d'intensité et de durée[KD06]. Bien que cette représentation permette d'extraire efficacement des informations musicales sur différents niveaux d'abstraction (mélodie, rythme, tempo, partition, style), elle rencontre des limites pour distinguer d'autres qualités locales, de même niveau que la note. Pour désigner ces unités locales de façon générique, nous employons ici le terme d'*événements musicaux*. La figure 1 donne des exemples de cette information musicale qui peut enrichir la notion de note.

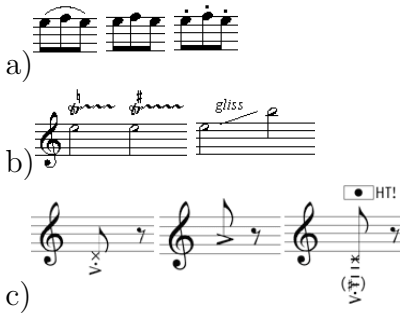

FIG. 1 – *Exemples d'événements musicaux : a)Articulations entre notes : legato, non-legato, staccato. b)Mouvements de hauteur : trilles, glissando. c)Modes de jeu (ici, pour la flûte traversière) : key-click, slap, tongue-ram.*

Bien que l'on puisse trouver une typologie de ces événements dans des ouvrages d'orchestration, ou dans certaines banques d'échantillons spécialisées, leur exhaustivité est compromise par le fait qu'ils peuvent être propres à un instrument (modes de jeu) ou inventés par le compositeur. Par exemple, plus d'une centaine de modes de jeu pour saxophone ont été répertoriés en examinant les combinaisons sur le champs des hauteurs, des nuances, des modes d'attaque et d'entretien [Kie90]. Le système visé ici ne prétend pas proposer une modélisation exhaustive de tous les événements possibles. Il est décrit par le scénario applicatif suivant : le musicien ou le compositeur définit et enregistre un ensemble d'événements lors d'une phase d'apprentissage supervisée, le système est chargé de reconnaître et segmenter (en temps réel et dans le flux audio) ces événements rejoués au cours d'une interprétation.

0.1.3 Portée de l'étude

La figure 2 montre les étapes principales de traitement du flux des donné dans le système visé :

FIG. 2 – Étapes de traitement

Notre travail porte sur le troisième bloc, c'est-à-dire de l'étude et de la mise en œuvre de méthodes d'apprentissage automatique pour la modélisation et la reconnaissance d'événements musicaux, tels que présentés au paragraphe précédent.

0.2 Littérature

Du point de vue de l'apprentissage statistique, un signal audio est une série temporelle de données, a priori non-indépendantes. Les modèles les plus répandus pour traiter les données séquentielles sont basés sur des chaînes de Markov. Ils sont utilisés aussi bien pour le décodage génomique que dans la plupart des systèmes actuels de reconnaissance automatique de la parole (par exemple HTK[1] et CMU Sphinx[2]), dans leur version à variable cachée dite *modèle de Markov caché* (HMM)

Dans le domaine de la transcription musicale, les HMMs sont employés avec un succès grandissant pour plusieurs types d'applications. Ils nous a donc semblé naturel de privilégier l'étude de ces modèles, afin d'en cerner les intérêts et les limites dans notre cas. Nous présentons brièvement quelques éléments de la littérature comme points d'appui pour exposer les problématiques que nous avons étudié jusqu'à présent.

0.2.1 HMMs : un bref rappel

Cette famille de modèles exprime la distribution jointe de probabilité d'une variable aléatoire *cachée* X et d'une variable aléatoire *observée* Y. Dans le cas d'un espace d'état discret de dimension K, X prend ses valeurs dans $\{x_1, ..., x_K\}$. On note $q_t = x_k$ *l'état* du système au temps t. Dans le cas d'un espace d'observation continu, Y prend ses valeurs dans $\mathbb{R}^D$. Un modèle est paramétré par :

- un vecteur π de dimension K : c'est la distribution de X pour $t = 0$.
- une matrice A de dimension $[K \times K]$: chaque élément $a_{i,j}$ est la probabilité de transition de $q_{t-1} = S_i$ à $q_t = S_j, \forall t$. L'ensemble des $a_{i,j}$ non-nuls définit la *topologie* de A.
- une loi de probabilité dite *d'émission* ou *d'observation* qui représente la distribution de probabilité conditionnelle $P(Y|X)$ de la variable d'observation Y connaissant l'état X.

Modélisation : On appelle *structure* du modèle, le nombre d'états du modèle K, la topologie de A et la loi $P(Y|X)$ choisie. Un HMM est un modèle dit *génératif*, dans le sens où il représente des données par une densité de probabilité, qui approxime la distribution réelle de sa classe de données sur l'espace d'état et d'observation choisis. Ainsi, un état peut s'interpréter comme l'une des phases temporelles de la séquence apprise. Cette correspondance permet d'introduire une connaissance *experte* des données pour

[1] http ://htk.eng.cam.ac.uk/
[2] http ://cmusphinx.sourceforge.net/html/cmusphinx.php

proposer la structure d'un modèle, voire la valeur de ses paramètres.
Optimisation : Les valeurs des paramètres sont estimés à partir des données d'un *corpus d'apprentissage* contenant des représentants de la classe du modèle considéré.
Reconnaissance : La reconnaissance passe par une étape de *décodage* qui met en correspondance les trames successives de données en états du modèle.

0.2.2 Modèles de séquences de notes

Les HMMs sont modulaires dans le sens où l'on peut regrouper des modèles de bas niveau en modèles de niveau supérieur. En reconnaissance automatique de la parole, si l'unité acoustique de base choisie est le phonème, un modèle de mot est construit par concaténation des modèles de phonèmes [Rab89].

Ce même principe peut être appliqué pour représenter des séquences de notes. Par exemple dans [Rap99], la partition est représentée par une concaténation de modèles de notes : sur la figure 3.a chaque état représente l'une des phases de la note. On atteint un niveau d'abstraction supérieur en considérant une nouvelle chaîne dans laquelle chaque événement musical (note d'une hauteur donnée ou silence) représente un état. Cela permet de modéliser des aspects tels que la probabilité d'une transition entre deux notes dans un contexte musical connu grâce à un modèle musicologique [RK04], (figure 3.b), ou encore la probabilité d'une erreur d'interprétation en un point délicat de la partition [OD01] (figure 3.c)

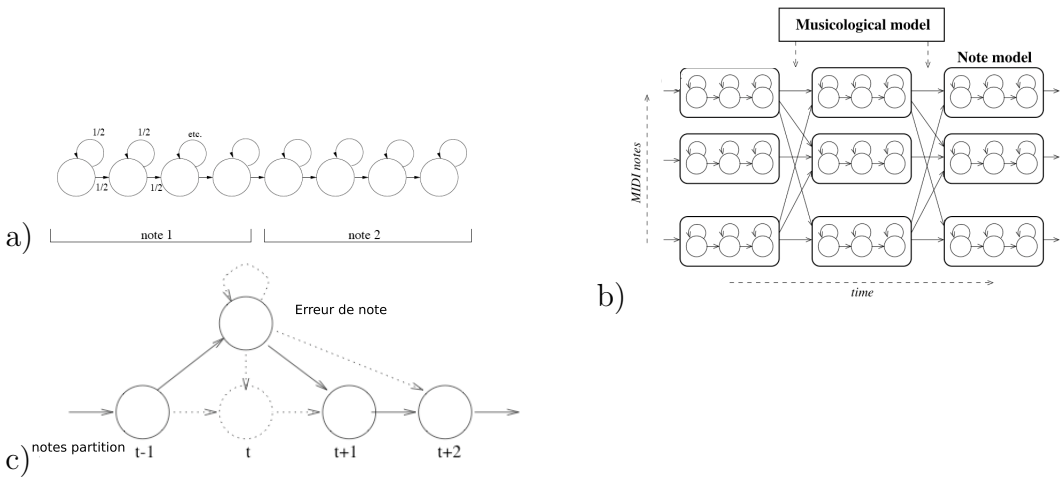

FIG. 3 – *Exemples de modèles de séquences de notes. a)deux notes enchaînées d'une partition, b)modèle musicologique qui contrôle les probabilités de transitions entre notes, c)modélisation d'une erreur d'interprétation en suivi de partition.*

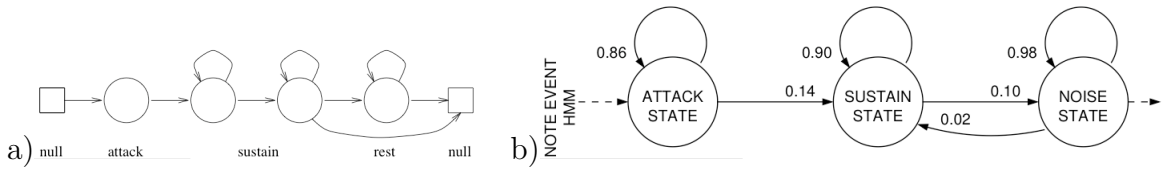

FIG. 4 – Modèles acoustiques

0.2.3 Modèles acoustiques

Les unités acoustiques élémentaires sont modélisées par un ensemble d'états, chacun représentant une distribution sur les variables d'observations du modèle. Cette distribution est le plus souvent modélisée par un mélange de Gaussiennes (GMM). Pour la modélisation de notes, une relation entre la structure temporelle d'une note et le modèle est souvent explicitée dans le modèle pour représenter une connaissance a priori (fournie par *l'expert*) sur les unités représentées. Par exemple, dans [OD01], une note est représentée par un modèle gauche-droit (figure 4.a), qui suppose qu'une note observée passe par une courte phase d'attaque (une seule trame), une phase de son entretenu -*sustain*- plus longue, et termine par un silence -*rest*- optionnel. Dans une application de transcription [RK05b], on trouve un modèle de note assez semblable, mais dans lequel l'attaque peut être prolongée, et le son entretenu entrecoupé de phases de bruit en fin de note (figure 4.b).
Les descripteurs instantanés choisis doivent être cohérents avec le modèle, c'est-à-dire tels que la densité d'observation de chaque état possède des modes suffisamment distincts pour minimiser la confusion possible entre états. Nous n'énumérerons pas ici les descripteurs utilisés dans la littérature, mais on peut néanmoins les distinguer en plusieurs catégories [Pee04] :

1. descripteurs temporels
2. descripteurs de l'énergie
3. descripteurs spectraux
4. descripteurs harmoniques
5. descripteurs perceptifs

On notera qu'il existe peu de littérature sur la sélection automatique de descripteurs pour les HMMs. Ils sont le plus souvent choisis empiriquement. Dans le domaine de la parole, la question est rarement abordée étant-donnée l'efficacité des coefficients MFCC (*mel frequency ceptral coefficients*) et de leurs dérivées premières et secondes.

0.2.4 Optimisation

L'apprentissage des paramètres est obtenu en minimisant (ou maximisant, le cas échéant) un critère d'optimalité, exprimé par une fonction d'objectif dépendant des paramètres du modèle et d'un corpus d'apprentissage constitué d'un ensemble de N séquences $\mathbf{O}_{\mathbf{T_n}}^{(\mathbf{n})}$ de durées T_n, pour chaque classe $m \in \{1..M\}$.
Le critère le plus répandu est celui du *maximum de vraisemblance*, qui fait tendre les paramètres de la densité du modèle au plus proche de la distribution empirique de l'échantillon d'apprentissage. Il n'existe pas de méthode analytique pour résoudre ce problème. En revanche, la méthode itérative EM (*expectation-maximization*) pour les HMMs (algorithme de *Baum-Welch*) est efficace, pour peu que l'initialisation du modèle soit suffisamment proche de la solution globale, et que les données soient en nombre suffisant par rapport à complexité du modèle, afin d'éviter le sur-apprentissage [Gha01]. Il est également possible de choisir un critère plus directement relié à la tâche ultime de classification, en favorisant les paramètres qui s'accordent aux données de leur classe, tout en s'éloignant au plus des données des classes concurrentes. Cet apprentissage, appelé *discriminatif*, est basé sur l'optimisation d'un critère MCE (*minimum classification error*) [RD96], ou MMI (*maximum mutual information*) [Nor91]. L'utilisation de ces critères permet de réduire les taux d'erreur de classification [LRM05]. Notons que de récentes méthodes d'apprentissage discriminitatif permettent une formulation convexe de la fonction d'erreur à optimiser, évitant ainsi les solutions erronées dues à des minima locaux [SS07].

0.2.5 Reconnaissance et segmentation

Le décodage des trames d'observation en états du modèle est l'étape qui permet d'obtenir une information symbolique. Les états *cachés* du modèle sont inférés à partir d'une séquence d'observation en choisissant le chemin d'états le plus vraisemblable, étant-donné le modèle et la séquence $\mathbf{O_T}$. Le décodage du meilleur chemin permet d'aligner les états aux trames de signal, opérant de fait une segmentation et une reconnaissance. Un décodage optimal est obtenu avec l'algorithme de Viterbi [Rab89] qui utilise une méthode de programmation dynamique [Bel57]. Le principe de cette méthode consiste à mémoriser pour chaque instant et pour chaque état, l'état précédent q_{t-1} expliquant au mieux le chemin qui aboutit en q_t. Une fois que l'état le plus vraisemblable est identifié pour la dernière trame en T, le décodage de toute la séquence s'opère par itérations arrières.

0.3 Problématiques

Les problématiques sur lesquelles nous nous sommes concentré jusqu'ici sont explicitées.

0.3.1 Expert absent

Dans la section 0.2.3, nous avons vu deux exemples de la littérature où la modélisation des notes prend en compte une connaissance *a priori* sur la structure temporelle d'une note : celle d'un agent dit *expert* qui sait par expérience qu'une note peut être schématisée par une phase d'attaque, d'une partie entretenue de longueur variable et d'une phase optionnelle de silence.
Dans le scénario applicatif visé (introduit à la section 0.1.2), *l'expert* est volontairement absent puisque l'on veut éviter de faire un *catalogage* de modèles, dont l'exhaustivité serait sans-doute impossible. Au contraire, il s'agit de proposer un système qui apprend la structure des modèles sur des données d'apprentissage. Une problématique identifiée est donc celle de la modélisation sans expert.

0.3.2 Peu de données

Une conséquence implicite du scénario applicatif est que l'apprentissage devra se contenter de peu de données, puisque les classes apprises peuvent être très spécifiques à une œuvre, et donc ne pas exister dans d'autres bases de données. Nous devons donc prendre en compte cette contrainte dès la modélisation, en favorisant des modèles dont le nombre de paramètre est minimal.

0.3.3 Temps-réel

Il s'agit d'effectuer une reconnaissance dans un flux, le plus rapidement possible, pour qu'un situation d'intéraction musicale soit possible. Cela implique principalement une analyse causale des données, ainsi qu'une complexité limitée pour effectuer le décodage. Nous avons vu dans la section 0.2.5 que le décodage optimal se fait par itérations arrières en démarrant par la dernière trame observée, ce qui est anti-causal. Dans un contexte de temps-réel, il faut déterminer une méthode de décodage plus adaptée.

0.4 Travaux effectués

Démarche

Afin de ne pas multiplier dans un même temps les sources de difficulté, nous avons choisi de concentrer cette première phase de travail sur des données synthétiques, c'est-à-dire obtenues par échantillonage à partir de modèles connus, que nous appelons *générateurs*. Cela nous permet d'expérimenter en contrôlant finement les hypothèses de travail. Nous présentons ici les résultats de nos travaux préliminaires sur les problématiques identifiées plus haut.

Note sur l'environnement de travail

Une fois le cadre de modélisation choisi et les contraintes définies, nous avons écrit une bibliothèque de fonctions Matlab nous permettant de manipuler des HMMs, et de garder un contrôle précis sur les algorithmes d'optimisation et de décodage, et d'expérimenter. Nous utilisons la bibliothèque Netlab[Nab02] pour manipuler les GMMs. Afin d'avoir un exemple de référence et de comparaison quant à la mise en pratique, nous nous sommes familiarisé avec l'outil *HMM Toolkit* (HTK), très bien référencé mais spécialisé en reconnaissance de la parole, et bien que très paramétrable, il est difficile à modifier dans un contexte de recherche.
La base de donnée existante la plus proche de notre problème est sans-doute la base *Studio On Line* (SOL), qui collectionne environ 19400 fichiers sons de notes isolées à différentes hauteur jouant utilisant différents modes de jeu ou de dynamiques, pour 16 instruments différents. Elle ne comporte pas de phrases musicales utilisant ces modes de jeux.
Nous avons annoté une séance de travail de flûte traversière de la pièce *Quasikristall* du compositeur Hector Parra. Cette base est de petite taille (5 minutes d'audio) mais très précisément annotée sur plusieurs plans chevauchés (attaques, dynamiques, modes de jeu, enveloppe mélodique), et comporte plusieurs versions de phrases musicales avec des modes de jeu variés. En cela, elle correspond au type de données auxquelles le système devrait être confronté *in fine*.

0.4.1 Sélection de modèles

Dans cette section, nous présentons l'avancée de nos travaux quant aux deux premières problématiques présentées précédemment, à savoir la modélisation sans expert, et l'apprentissage sur peu de données. Les méthodes dites de *sélection de modèle* répondent aux deux problèmes en choisissant le modèle le plus parcimonieux qui explique au mieux les données, selon l'heuristique du *rasoir d'Occam*. Dans le contexte applicatif de la reconnaissance sur des données audio, la sélection de modèle est étudiée dans le domaine de la *reconnaissance de son générique*[3], qui tente de catégoriser des segments isolés de son, qui peuvent provenir de différentes sources (bruits de pas, animaux, instrument de musique) [Cas01],[RGE03]. Nous présentons ici une mise en application de la sélection de modèle pour la modélisation d'événements musicaux.

Heuristique de sélection

La sélection de modèle vise à choisir parmi un ensemble de structures de modèles, celles qui sont le plus adaptées à un problème donné, qui se présente sous la forme d'un ensemble de données et d'un critère d'optimisation. La structure d'un modèle désigne de manière générale une famille de modèles et le nombre de paramètres au sein de cette famille. En pratique, la sélection se fait parmi un jeu de modèles candidats qui représentent un ensemble de structures potentielles. Une heuristique de modélisation répandue (rasoir d'Occam) consiste à choisir la modélisation "la plus simple" qui représente les données, afin de pouvoir généraliser cette modélisation aux données à venir. En modélisation, cette notion de simplicité s'exprime à travers la *complexité* de la structure d'un modèle, soit le nombre de ses paramètres. Nous avons étudié deux méthodes appliquant cette heuristique de sélection : la *validation croisée* et le *critère d'information Bayésienne*.
Pour les GMMs, on peut appeler structure le nombre de Gaussiennes (ou centres), pour un choix fixé de type de covariance. La figure 5 montre l'évolution de l'erreur empirique sur deux jeux de données (apprentissage et test, obtenus par *validation croisée*) générés par le même modèle à trois centres. Le point de divergence entre les deux courbes indique en abscisse le nombre de centres au delà duquel les modèles sur-apprennent les données d'apprentissage, et ne sont pas capables de généraliser à de nouvelles données. Ce point permet de sélectionner la structure optimale pour le jeu de données correspondant. Ainsi, la complexité du modèle est maîtrisée. La validation croisée est cependant une méthode peu économique en temps de calcul, et d'autant moins que le jeu d'apprentissage est réduit (ce qui est le cas ici) [Bis06]. La sélection par *critère d'information Bayésienne* (BIC, voir l'annexe A pour une présentation) permet en comparaison de ne

[3]Generalized Sound Recognition

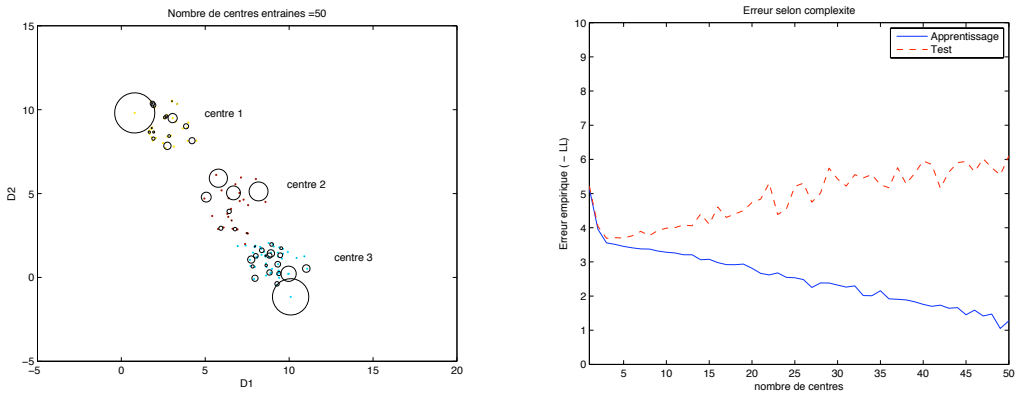

FIG. 5 – *A gauche, les données générées par un GMM à $J = 3$ centres (trois nuages de points), apprises par un modèle à $J = 50$ centres (cercles) : situation de sur-apprentissage. A droite, évolution de l'erreur en fonction de de la dimension du modèle entraîne sur les mêmes données générées par le modèle à trois centres.*

nécessiter qu'une passe d'apprentissage pour chaque structure envisagée. Ce critère est une approximation de la *vraisemblance intégrée*[4]. Cette dernière donne une mesure de la "préférence" des données pour une structure considérée.

Sélection de HMMs avec le critère BIC

Nous avons mis en pratique ce critère de sélection sur des ensembles de données synthétiques, en prenant comme hyperparamètres de structure le nombre d'états K et le nombre de composantes J du GMM pour un état. Les données sont générées à partir de M modèles λ_m de structures différentes, contrôlées par K et J. Pour chaque modèle, on génère $6N$ séquences d'apprentissage, $2N$ séquences de validation et $2N$ séquences de test. L'espace de recherche est borné par K_{max} et J_{max}, ce qui engendre $(K_{max} \times J_{max})$ modèles candidats. Ils sont entraînés par maximum de vraisemblance (cf. 0.2.4) sur le corpus d'apprentissage.
La valeur du BIC est paramétrée par α (cf. eq. A.3) qui contrôle la pénalisation de la complexité du modèle. La figure 6 montre les valeurs de BIC des modèles candidats pour une classe donnée, avec des valeurs croissantes de α. La sélection des modèles se fait pour un α donné, le jeu des modèles retenus peut alors être évalué sur une tâche de classification [Bie03].
Nous classifions les données du corpus de validation avec les jeux de modèles obtenus pour des valeurs croissantes de α. La figure 7.a montre les scores de classification entre 3 classes sur l'ensemble de validation. En choisissant le α qui maximise le score, on effectue

[4]appelée *evidence* dans la littérature anglophone

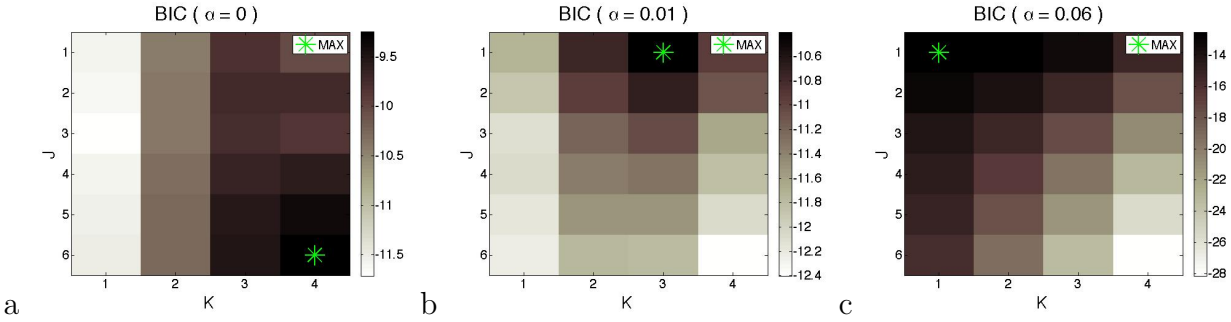

FIG. 6 – *Evolution du BIC sur le même modèle, pour des valeurs croissantes de α, sur l'espace des hyperparamètres K et J. a) lorsque α est nul, le critère favorise les modèles complexes qui sur-apprennent les données. b) et c) le BIC favorise les modèles plus simples pour α non nul.*

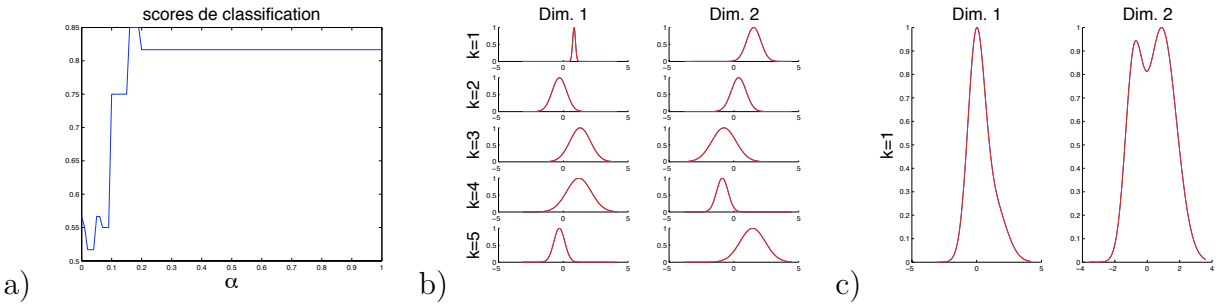

FIG. 7 – *Exemple de sélection de modèle : a)Scores de classification sur l'ensemble de validation, le maximum est obtenu pour $\alpha = 0.17$. b)États du modèle utilisé pour synthétiser les données d'une classe (un état par ligne, une dimension d'observation par colonne.) c)États du modèle sélectionné pour la même classe.*

de fait une sélection discriminative de modèles. Dans l'exemple présenté ici, on retient les modèles correspondant à $\alpha = 0.17$, et les figures 7.b et 7.c montrent respectivement l'un des modèles ayant généré les données d'une classe (avec $K = 5$ et $J = 1$), et le modèle sélectionné pour cette même classe ($K = 1$ et $J = 2$). Si l'on compare les deux modèles, on peut observer que le deuxième est une intégration temporelle du premier. Dans ce cas, le modèle sélectionné est un HMM à un seul état (la réduction en complexité est mise en évidence).

Une fois le paramètre α déterminé, un nouvel apprentissage (par méthode EM) des modèles sélectionnés est effectué, après avoir injecté les données de validation dans l'ensemble d'apprentissage. Les modèles obtenus sont alors évalués par la tâche de classification sur les données de l'ensemble test. Des résultats obtenus pour le même exemple sont donnés dans le tableau 1. On peut constater que les scores, bien que sous-optimaux, sont relativement proches entre modèles générateurs et modèles sélectionnés.

a)

%	**c1**	**c2**	**c3**
c1	60	25	15
c2	30	70	0
c3	10	0	90

b)

%	**c1**	**c2**	**c3**
c1	85	15	0
c2	15	85	0
c3	35	10	55

TAB. 1 – *Exemple de scores de classification pour 3 classes, avec $N = 10$ (soit 60 séquences test au total.) a)Matrice de confusion obtenue avec les modèles sélectionnés (taux de reconnaissance = 73.33%). b)Matrice de confusion avec les modèles utilisés pour la synthèse (taux=75%).*

En choisissant des états plus dissemblables pour les modèles générateurs, on a pu observer des scores de reconnaissance plus élevés pour les modèles sélectionnés automatiquement que pour les modèles générateurs. La figure 8 montre un tel cas : la classification par trois modèles (avec $N = 50$, soit 300 séquences test au total) donne un score de 59.33% pour les modèles générateurs, contre un score de 99% avec les modèles sélectionnés par le critère BIC. Ces derniers possèdent toujours un nombre d'états inférieur à celui des modèles générateurs, et qui tend souvent vers une intégration temporelle dans un modèle à un seul état.

En résumé

Nous avons mis en application un principe de sélection de modèle par critère BIC sur des données synthétiques, observé qualitativement le résultat de cette sélection sur les structures obtenues (réduction d'états) et observé quantitativement de bons scores de classification, surtout lorsque les états des modèles sont plus dissemblables. Malgré la réduction de complexité et une bonne généralisation des modèles obtenus, il se pourrait que cet "aplatissement" de la structure temporelle du modèle se fasse au détriment de la résolution temporelle de la segmentation, présentée dans la section suivante.

0.4.2 Reconnaissance en temps réel

Nous présentons ici des mises en œuvre pratiques de segmentation et de reconnaissance avec les HMMs, ainsi qu'une piste pour effectuer un décodage optimal dans un contexte de temps réel.

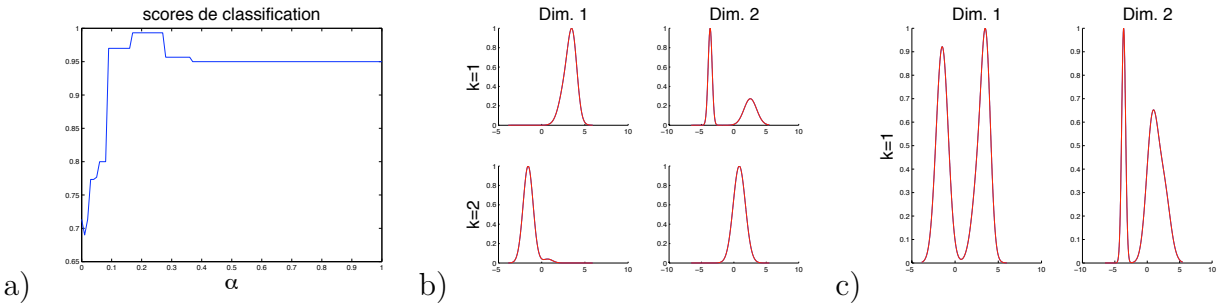

FIG. 8 – *Sélection de modèle, avec des états plus distincts : a)Scores de classification sur l'ensemble de validation. b)États du modèle générateur d'une classe. c)États du modèle sélectionné pour la même classe.*

Principe de segmentation par Viterbi, en temps différé

Nous avons étudié une modélisation au seul niveau acoustique, c'est-à-dire avec des transitions équiprobables entre chaque modèle acoustique élémentaire. On garde à l'esprit que dans une application ciblée pour un contexte donné, les probabilités de transition entre modèles acoustiques peuvent être pris en charge par un modèle de plus haut niveau (modèle de langage dans le cas de la parole, modèle de partition dans le cas de suivi de partition, ou modèle musicologique dans le cas de transcription) qui contribue à améliorer les performances de reconnaissance.
La structure du modèle sur lequel nous travaillons est construite par la concaténation de M modèles acoustiques à K_m états, dont les densités d'observation sont modélisées par des GMMs (figure 9.a). Le modèle global compte alors $K_M = \sum_{m=1}^{M} K_m$ états. La correspondance entre les indices des états du modèle global et ceux des modèles acoustiques étant mémorisée, un décodage sur le modèle global permet de retrouver la séquence des modèles acoustiques la plus vraisemblable étant donnée une séquence d'observation $\mathbf{O_T}$. La figure 9.b montre, un décodage Viterbi sur une séquence d'observation synthétique générée par la concaténation de 5 modèles. Cette séquence d'états optimale peut être "repliée" en une séquence de modèles, et chaque saut d'état de la fin d'un sous-modèle au début d'un nouveau constitue un nouveau point de segmentation (figure 9.c).

Décodage par variable avant

Le décodage optimal obtenu par l'algorithme de Viterbi fonctionne par retour-arrière à partir de l'instant T sur une structure mémorisant les transitions locales les plus probables à chaque instant t de la séquence $\mathbf{O_T}$ de longueur T. Si l'on ne dispose des

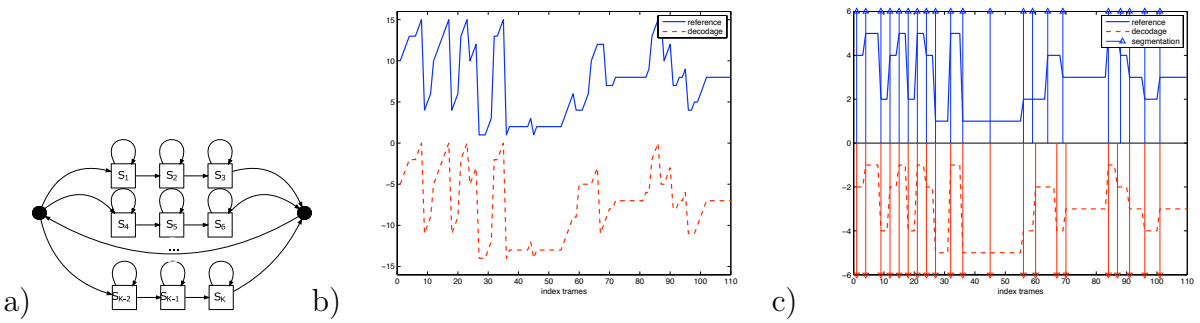

FIG. 9 – a)*Exemple d'un modèle global construit par concaténation de sous-modèles à trois états chacun*, b)*Décodage d'états du modèle global*, c)*Segmentation résultante.*

observations que jusqu'à un instant t, cette méthode n'est pas applicable. En suivi de partition, la méthode utilisée consiste utiliser la *variable avant*, souvent notée $\alpha_t(i)$ dans le formalisme des HMMs. Cette variable représente la probabilité de l'observation partielle de O_1 jusqu'à O_t et de l'état S_i au temps t, connaissant le modèle λ. Le décodage consiste à déterminer l'état q_t le plus probable à l'instant t [Con04] :

$$q_t = argmax_{1 \leq K}[\alpha_t(i)] \tag{1}$$

Le décodage obtenu n'est pas optimal, car il se peut qu'à l'instant $t+1$, l'état le plus probable ne soit pas permis par la topologie du modèle. Ainsi, il faut connaître les observations suivantes pour décider de l'état optimal en t. En nous inspirant de travaux récents sur l'alignement temporel, nous avons élaboré un algorithme de décodage qui utilise une fenêtre d'observation à court terme pour effectuer un décodage optimal local. Nous en présentons le principe dans les paragraphes suivants.

Décodage Viterbi à court-terme

Le *Dynamic Time Warping* (DTW) est un algorithme qui permet de trouver l'alignement temporel optimal entre deux séquences de données dont le contenu est proche mais, mais dont l'écoulement temporel est asynchrone. Cet algorithme peut être appliqué à l'alignement entre une partition et son interprétation audio. Appliqué sur l'intégralité des séquences, il permet de retrouver le meilleur chemin d'alignement dans la matrice des distances locales entre tous les instants des deux séquences. En pratique, la nécessité de travailler sur l'intégralité des séquences limite la longueur des séquences alignables, pour cause de ressource mémoire disponibles, et de temps de calcul. L'algorithme *Short-Time Dynamic Time Warping* (STDTW) permet d'obtenir le même chemin optimal que par DTW, mais de manière itérative, sur des portions successives des séquences [KR06].
L'algorithme DTW repose sur la minimisation d'une distance cumulée. En utilisant le

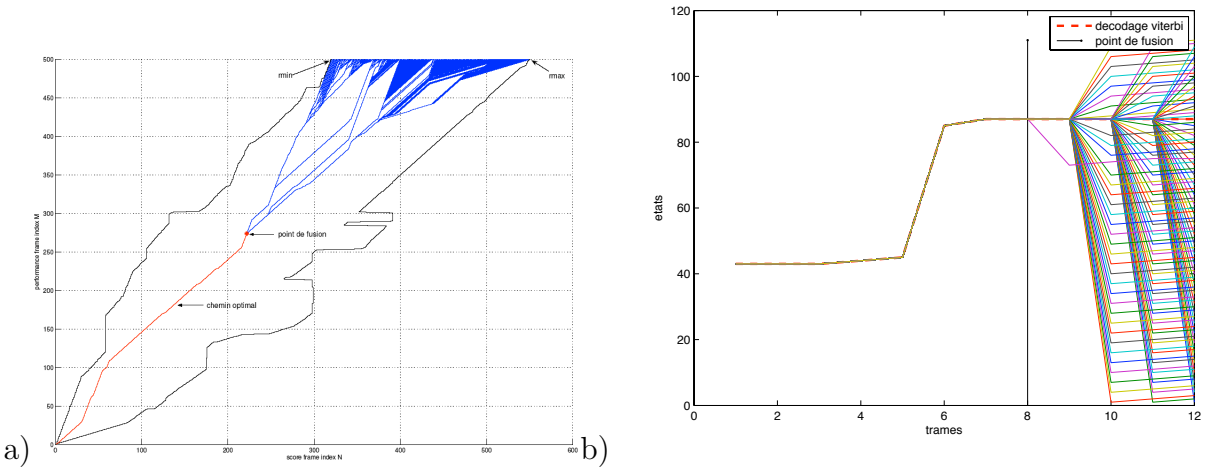

FIG. 10 – *Points de fusion : a)Point de fusion DTW [KR06] b)Point de fusion des chemins sur un treillis d'états : Le chemin décodé par Viterbi global est superposé en pointillés.*

principe d'optimalité locale, la programmation dynamique permet de trouver le chemin global qui minimise cette distance, en mémorisant les distances locales. L'algorithme de Viterbi pour le décodage des états cachés d'un HMM applique également ce principe d'optimalité, en calculant itérativement la variable $\delta_t(i)$, qui est la probabilité du chemin le plus vraisemblable aboutissant à l'état i en l'instant t.
Le passage de DTW à sa version court terme repose sur l'observation d'une portion commune entre tous les chemins les plus probables pour un ensemble de points d'alignement donnés. Ces chemins sont confondus en un *point de fusion* comme le montre la figure 10.a.

En transposant l'idée à un treillis d'états de HMM sur une fenêtre d'observation locale suffisamment large, nous avons pu observer un point de fusion de même nature entre tous les chemins aboutissant aux K_M états d'un modèle global (cf. section 0.4.2) en l'instant t_D du bord droit de la fenêtre (figure 10.b). Nous avons alors implémenté un algorithme de décodage, qui opère localement sur des fenêtres successives, et dont le principe est le suivant :

1. initialisation des bords de la fenêtre ($t_G = 0$ et $t_D = 1$).
2. incrémenter t_D
3. s'il n'y a aucun point de fusion dans la fenêtre d'observation, calculer par Viterbi les chemins optimaux terminant aux K_M états en t_D. Sinon, mémoriser le chemin jusqu'à l'instant t_{PF} du point de fusion et remplacer $t_G = t_{PF}$
4. tant que $t \leq T$, retour à l'étape 2, sinon terminer le décodage.

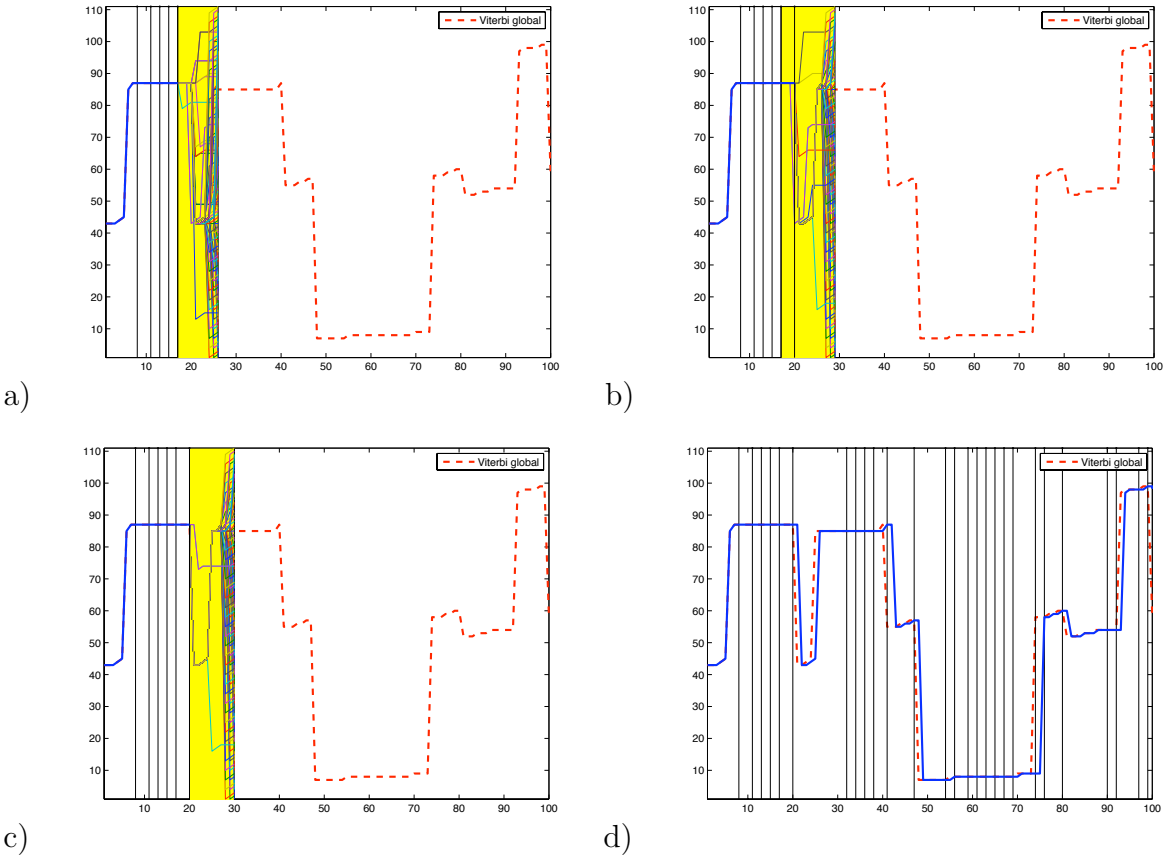

FIG. 11 – *Étapes de décodage Viterbi à court terme. En pointillés : décodage Viterbi global obtenu en temps différé. En trait plein : décodage obtenu par Viterbi à court terme. Zone grisée : fenêtre locale. Traits verticaux : points de fusion. a)Tous les chemins dans la fenêtre aboutissent au bord gauche : pas de point de fusion. b)Un point de fusion est trouvé dans la fenêtre élargie. c)Le bord gauche de la fenêtre est décalé au dernier point de fusion trouvé. d)Résultat final.*

La figure 11 illustre ce fonctionnement. On peut voir que le chemin obtenu par Viterbi à court terme est très semblable au décodage par Viterbi global, bien qu'un petit décalage subsiste, qui nécessite une investigation sur ce point.

En résumé

Nous avons proposé un algorithme de décodage Viterbi à court-terme, pertinent dans un contexte de temps-réel pour une tâche de segmentation et de reconnaissance. Il permet d'obtenir un décodage quasi-optimal (l'optimalité est visée), avec un délai de décision dépendant de la taille de fenêtre nécessaire à la détection d'un point de fusion.

0.5 Perspectives

Nos résultats préliminaires sur la sélection de modèle et la reconnaissance en temps réel sont encourageants. Ils suscitent toutefois des possibilités d'amélioration, et des idées nouvelles d'expérimentation, sur les points suivant :

Sélection de modèles : Nous avons pu observer qualitativement qu'en choisissant des états plus "dissemblables" pour les modèles générateurs, les modèles sélectionnés donnent de meilleures performances de classification. Ceci laisse penser qu'un apprentissage discriminatif sur les modèles candidats (au lieu de l'apprentissage par EM actuellement en place) pourrait améliorer le système. De plus, certaines de ces techniques permettent d'attribuer un nombre variable de Gaussiennes à chaque état du modèle [Nor95], ce qui pourrait contribuer à réduire encore la complexité.
Par souci de progression dans la difficulté, nous n'avons pas encore intégré le choix de la topologie dans l'espace des hyperparmètres, en imposant toujours des modèles gauche-droits. Il paraît essentiel de faire sauter cette contrainte, surtout dans les cas où les événements musicaux modélisés comportent des cycles répétés (ce qui est le cas pour une *trille* par exemple).

Segmentation par Viterbi à court terme : Les étapes suivantes pour améliorer cette idée, consistent à valider le caractère optimal du décodage obtenu, et à évaluer quantitativement les performances de l'algorithme sur des tâches de classification, et de segmentation.
Un aspect à examiner plus particulièrement concerne le temps de délai pour le décodage. Nous avons vu que pour l'instant, il est variable. Il s'agit d'une part d'évaluer quantitativement les valeurs de ces délais pour différentes combinaisons de modèles, et d'autre part, d'étudier les paramètres qui permettraient de donner une borne supérieure à ce délai. Cette *latence* est en effet un paramètre crucial dans toute application temps-réel.

Combinaison des deux pistes : Nous avons observé que la sélection de modèle avec le critère BIC à une forte tendance à écraser la structure temporelle des modèles sélectionnés, en favorisant les modèles à peu d'états. Bien que performants pour une tâche de classification sur des unités isolées, il paraît vraisemblable que ces modèles donneront des résultats peu précis en matière de segmentation temporelle. Si cette hypothèse est vérifiée, nous pourrons modifier notre algorithme de sélection de modèle, en élaborant un critère de sélection qui prenne en compte les performances de segmentation, et les minima en latence de décodage.

Application aux données réelles : Nous prévoyons d'appliquer les idées développées aux données réelles présentées au début de la section 0.4. De plus, une collaboration sui-

vie est prévue avec le compositeur Marco Stroppa, afin de travailler sur un ensemble d'événements musicaux dans ses pièces existantes, qui utilisent déjà un suivi de partition limité au suivi des hauteurs de notes. Une mise en application du travail de thèse pourrait alors être concrétisée dans le contexte qu'il vise.

Bibliographie

[ABC+06] G. Assayag, G. Bloch, M. Chemillier, A. Cont, and S. Dubnov. Omax brothers : A dynamic topology of agents for improvisation learning. In *ACM Multimedia Workshop on Audio and Music Computing for Multimedia*. Santa Barbara, October 2006.

[BDA+05] J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M.B. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13 :1035– 1047, 2005.

[Bel57] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[Bie03] A. Biem. A model selection criterion for classification : Application to hmm topology optimization. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, 2003.

[Bis06] C.M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[Cas01] M. Casey. Reduced-rank spectra and minimum-entropy priors as consistent and reliable cues for generalized sound recognition. In *Eurospeech*, 2001.

[Con04] A. Cont. Improvement of observation modeling for score following. Master's thesis, Université Pierre et Marie Curie, PARIS VI, 2004.

[Ess05] S. Essid. *Classification automatique des signaux audio-fréquences : reconnaissance des instruments de musique*. PhD thesis, Université Pierre et Marie Curie, 2005.

[Gha01] Z. Ghahramani. An introduction to hidden markov models and bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15 :9–42, 2001.

[KD06] A. Klapuri and M. Davy. *Signal Processing Methods for Music Transcription*. Springer, 2006.

[Kie90] D. Kientzy. *Saxologie*. PhD thesis, Université de Paris 8, 1990.

[KR06] H. Kaprykowsky and X. Rodet. Globally optimal short-time dynamic time warping, application to score to audio alignment. In *ICASSP 2006 Proceedings*, 2006.

[LRM05] J. Le Roux and E. McDermott. Optimization methods for discriminative training. In *Eurospeech*, 2005.

[Nab02] I.T. Nabney. *Netlab : Algorithms for Pattern Recognition*. Springer, 2002.

[Nor91] Y. Normandin. *Hidden Markov Models, Maximum Mutual Information Estimation and the Speech Recognition Problem*. PhD thesis, McGill University, 1991.

[Nor95] Y. Normandin. Optimal splitting of hmm gaussian mixture components with mmie training. In *IEEE ICASSP*, 1995.

[OD01] N Orio and F Déchelle. Score following using spectral analysis and hidden markov models. In *ICMC*, Havana, Cuba, 2001.

[Pee04] G. Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. Technical report, IRCAM, 2004.

[Rab89] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, 1989.

[Rap99] C. Raphael. Automatic segmentation of acoustic musical signals using hidden markov models. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 21, pages 360–370, 1999.

[RD96] C. Rathinavelu and L. Deng. The trended hmm with discriminative training for phonetic classification. In *ICSLP*, 1996.

[RGE03] M.J. Reyes-Gomez and D.P.W. Ellis. Selection, parameter estimation, and discriminative training of hidden markov models for general audio modeling. In *ICME-03*, pages I–73–76, Baltimore, July 2003.

[RK04] M. Ryynänen and A. Klapuri. Modelling of note events for singing transcription. In *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, 2004.

[RK05a] M. Ryynänen and A. Klapuri. Polyphonic music transcription using note event modeling. In *Proc. 2005 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 319–322, New Paltz, New York, USA, October 2005.

[RK05b] M. Ryynänen and A. Klapuri. Polyphonic music transcription using note event modeling. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'05)*, 2005.

[SCS05] D. Schwarz, A. Cont, and N. Schnell. From boulez to ballads : Training ircam's score follower. In *International Computer Music Conference (ICMC)*, Barcelona, Spain, Septembre 2005.

[She98] E. Sheirer. Tempo and beat analysis of acoustical musical signals. *Journal of the Acoustical Society of America*, 103 :588–601, 1998.

[SS07] F. Sha and L.K. Saul. Large margin hidden markov models for automatic speech recognition. In *Advances in Neural Information Processing Systems 19*, 2007.

[Vin04] E. Vincent. *Modèles d'instruments pour la séparation de sources et la transcription d'enregistrements musicaux.* PhD thesis, Université Paris VI, 2004.

Annexe A

Critère d'information Bayésienne

Nous rappelons brièvement la formulation du critère d'information Bayésienne (BIC).

On peut formaliser le problème de la sélection de modèle de la façon suivante : Pour chaque classe C_i à modéliser, il y a L_i modèles candidats $\{M_{il} : l = 1, ..., L_i\}$. Un modèle est défini par la combinaison d'une *structure* τ_{il} et d'un jeu θ_{il} de *paramètres*. Un critère de sélection $\mathcal{C}(.)$ est tel que pour chaque classe C_i, on choisit $\tau_{il} = argmax_{tau_{il}}\mathcal{C}(\tau_{il})$.

L'approche Bayésienne pour le choix de $\mathcal{C}$ consiste à maximiser la distribution jointe de τ_{il} et θ_{il} :

$$\mathcal{C}(\tau_{il}) = logP(X_i|\tau_{il}) + logP(\tau_{il}) \tag{A.1}$$

L'absence d'expert impose un *a priori* uniforme sur τ_{il}. Il en suit qu'il faut alors maximiser uniquement le premier terme de :A.1, soit la *vraisemblance intégrée* :

$$P(X_i|\tau_{il}) = \int p(X_i|\tau_{il}, \theta_{il})p(\theta_{il}|\tau_{il})d\theta_{il} \tag{A.2}$$

En pratique, cette intégrale n'est pas calculable. Une approximation possible est donnée par le BIC, selon la formule suivante :

$$logP(X_i|\tau_{il}) \approx BIC(\tau_{il}) = logp(X_i|\tau_{il}, \hat{\theta}_{il}) - \alpha\frac{K_{il}}{2}logN_i \tag{A.3}$$

où K_{il} est le nombre de paramètres de τ_{il}, N_i est la taille de l'échantillon X_i et α est un terme de régularisation.