Paramétrisation adaptée de transitoires pour la reconnaissance d'instruments de musique

Pierre LEVEAU

Mémoire de stage de DEA ATIAM année 2003-2004
Mars - Juillet 2004
Université Pierre et Marie Curie - Paris

Laboratoire d'Acoustique Musicale - ENST (Télécom Paris)

Encadrement : Laurent Daudet (LAM) - Gaël Richard (ENST)

Résumé

La reconnaissance automatique des instruments de musique s'appuie sur le calcul de paramètres bas-niveau sur les signaux audio à analyser. Choisir des paramètres adaptés consiste à prendre ceux permettant de discriminer efficacement les instruments. Ainsi, l'étude présentée a pour but d'ajouter des paramètres pertinents au corpus de paramètres utilisé pour la reconnaissance automatique d'instruments de musique sur des performances solo. L'idée conductrice est d'opérer une segmentation de ces solos en notes, puis en partie transitoire (*transient*) / parties tenues (*release*), dans le but de définir un jeu de paramètres adapté pour chaque type de segment. En effet, les transitoires sont censés contenir une grande partie de l'information lors de l'identification d'instruments de musique par l'homme.

Tout d'abord, des méthodes de détection de débuts de note (*onsets*) ont été développées, puis évaluées sur une base de donnée d'*onsets* de référence selon une méthodologie précisément définie. La méthode la plus adaptée aux besoins de l'algorithme de reconnaissance des instruments a ainsi été choisie, présentant de meilleurs résultats que les méthodes classiques. Puis des paramètres *a priori* adaptés aux transitoires ont été définis, ensuite testés pour la reconnaissance des instruments. Enfin, afin de confirmer sa validité, la méthode de détection d'*onsets* a été appliquée à l'extraction automatique de tempo.

Remerciements

Je tiens à remercier tout particulièrement mes maîtres de stage, Laurent Daudet (LAM) et Gaël Richard (ENST), pour m'avoir encadré et aidé, notamment pour ma première expérience de rédaction d'un article. Un grand merci également à Slim Essid et Miguel Alonso, thésards à l'ENST, pour leur précieuse collaboration concernant les applications de mes travaux.

Merci également à Jean-Dominique Polack pour m'avoir accueilli dans son laboratoire.

Enfin, je tiens aussi à remercier tous les stagiaires (particulièrement mes collègues David et Adrien) et permanents des deux laboratoires qui ont contribué au déroulement agréable de ce stage.

Table des matières

Table des figures

Introduction

L'indexation audio

Les données multimédia nécessitent une description associée pour un bon nombre d'applications : recherche de contenu, lecteurs multimédia, organisation de librairies, etc.. Les descripteurs *haut-niveau* comme le nom de l'artiste, de la chanson, le style, le tempo peuvent être relevés par un utilisateur. Cependant, l'accroissement considérable du volume de données multimédia disponibles en ligne ainsi que chez chacun des utilisateurs demande une automatisation de ces descriptions. L'indexation audio est le domaine de recherche développant les moyens de ces applications. Les processus mis en oeuvre se basent donc sur des calculs de paramètres (descripteurs *bas-niveau*) à partir du flux numérique, codé ou décodé. C'est à partir de ces paramètres qu'on peut déduire plus ou moins facilement des descripteurs haut-niveau qui font sens pour les utilisateurs.

Ce domaine de recherche est développé selon plusieurs axes à l'ENST : la détection de rythme et de tempo, la classification en segments sonores selon le contenu, la reconnaissance d'instruments de musique.

La reconnaissance automatique des instruments de musique

La reconnaissance automatique d'instruments de musique a plusieurs applications : elle permet d'obtenir des données intéressantes pour la description de fichier musicaux (ex : indications sur le style, l'époque d'un morceau), elle pourrait aboutir également à leur transcription automatique. Ce sujet de recherche est corrélé avec le traitement de la parole. La voix possède plus de possibilités de modulation qu'aucun instrument. Ainsi les méthodes utilisées pour le traitement automatique de la parole, domaine de recherche très développé étant donné ses nombreuses applications, sont d'une aide précieuse à la reconnaissance des instruments de musique.

La reconnaissance automatique d'instruments de musique est un problème relativement complexe. S'il devient possible de reconnaître des instruments sur des notes isolées avec des taux de reconnaissance acceptables (70% pour des instruments, 90% pour l'appartenance à une famille [2]), la tâche devient compliquée lorsqu'une phrase complète est analysée. Il est encore plus difficle de pouvoir distinguer plusieurs instruments dans des signaux multi-instruments.

Tout d'abord, l'idée de revenir à des notes isolées à partir de solos ou de mélanges d'instruments est un domaine de recherche en soi. Sur les phrases solo, nous verrons par la suite qu'il est difficile pour un sujet humain d'isoler les notes pour certains types d'instruments et certaines conditions d'enregistrement. Ceci rend l'évaluation des segmentations automatique des notes relativement peu fiables dans le cas général, ce que nous détaillerons plus tard. Si on étend le cadre de l'application aux signaux multi-instruments,

il s'agit alors d'effectuer une séparation de sources, puis d'effectuer une reconnaissance sur chacun des instruments séparés. Cependant la séparation de source demeure un problème au moins aussi complexe. Ainsi, si l'idée de revenir à des notes isolées est la plus simple du point de vue conceptuel, sa réalisation est extrêmement compliquée. Il s'agit donc d'utiliser des approches moins intuitives, mais techniquement plus réalisables et efficaces.

Pour réaliser une reconnaissance automatique d'instruments de musique, il faut tout d'abord calculer un certain nombre de paramètres sur le signal musical (ex : paramètres temporels, spectraux, hybrides...). Par comparaison avec ceux calculés pour des signaux dont on connaît l'instrument interprété, on cherche à obtenir l'instrument qui a joué la note. Pour cette décision, des méthodes statistiques complexes sont mises en jeu, faisant notamment intervenir un apprentissage des paramètres sur une base de données de sons labélisés. Dans ce type d'approche, un problème important à régler est le choix des paramètres pertinents. L'étude qui va suivre a pour objectif une paramétrisation adaptée à la reconnaissance d'instruments de musique sur des performances solo. L'approche choisie vise à différencier le choix des paramètres selon la partie du signal analysée, et mettra donc en jeu une segmentation des signaux musicaux.

Plan

Tout d'abord, nous allons évoquer les problématiques liées à la reconnaissance des instruments de musique, aussi bien relatives à la perception humaine qu'à la classification automatique. En conséquence, nous définirons le cadre et les objectifs de l'étude, notamment toute l'attention à porter sur la segmentation automatique de phrases musicales. Après avoir rappelé quelques éléments théoriques nécessaires à sa mise en oeuvre, nous décrirons les méthodes choisies puis leurs résultats relatifs à notre base d'évaluation. Après avoir choisi les paramètres adaptés à cette segmentation, nous évaluerons ce qu'elle apporte à la reconnaissance d'instruments de musique. Une application annexe à l'extraction du tempo sera également évoquée.

Première partie
État de l'art et rappels théoriques

1 Préliminaires

1.1 La reconnaissance des instruments : classification humaine et classification automatique

Évoquer les théories sur la classification d'instruments de musique par l'homme peut éclairer notre recherche de la paramétrisation adaptée pour la reconnaissance automatique.

1.1.1 La catégorisation des instruments par l'être humain, la perception du timbre

Les sujets humains reconnaissent les instruments de musique à leur "timbre". Cette notion est en relation avec un processus cognitif effectuant une catégorisation d'ensembles de paramètres perceptifs. Selon Helmoltz[1], on a longtemps défini la notion de timbre par la négative :"*on était généralement porté à attribuer au timbre, toutes les particularités des sons qui ne dérivaient pas directement de leur intensité et de leur hauteur*". Cependant il évoque que "*quelques unes de ces particularités dépendent de la façon dont les sons commencent et finissent*" : il remarque donc que la reconnaissance d'un instrument est lié aux transitoires d'attaque et d'extinction. Ces deux caractéristiques renvoient au mode de production du son, on parle alors de *timbre causal*. Du point de vue cognitif, c'est la première identification que nous mettons en oeuvre : cette écoute causale est quasi instinctive et très rapide. Les autres caractéristiques timbrales d'un son que nous pouvons distinguer sont liées à la qualité sonore, et demandent quant à elles plus de temps et de concentration.

Pour un même instrument, les variations spectrales sont extrêmement importantes d'une note à l'autre. On peut remarquer ceci lorqu'on observe des spectrogrammes. Cependant, nous sommes capables de trouver un invariant, commun à tous les sons de l'instrument : c'est ce qu'on définit par son timbre. Un grand nombre de perturbations n'altèrent pas la reconnaissance des instruments de musique par les être humains : les conditions d'enregistrement, de reproduction sonore, les mouvements de la tête, les déplacements par rapport à la source. En effet, un processus cognitif assure une certaine permanence de l'identification de la source.

Du point de vue signal, le timbre causal d'un instrument s'appuie sur des variations temporelles des caractéristiques spectrales. Celles-ci ont lieu au niveau des transitoires. L'évolution temporelle de l'enveloppe de l'énergie et

[1] dans *Théorie physiologique de la musique fondée sur l'étude des sensations auditives*

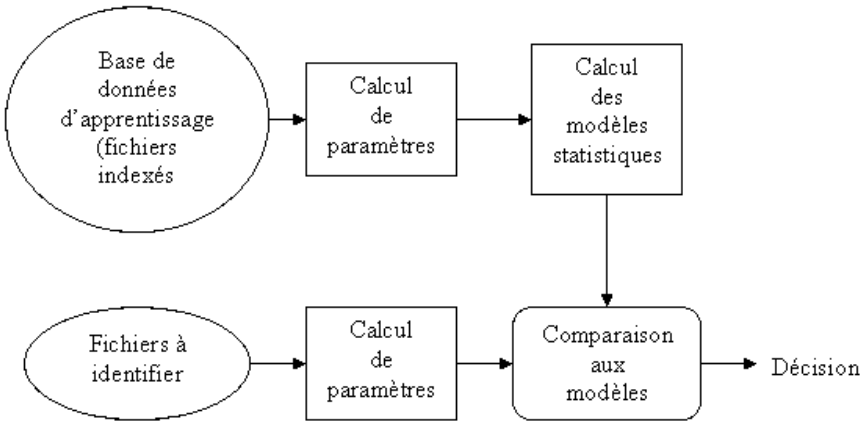

FIG. 1 – Schéma simplifié d'une reconnaissance automatique des instruments de musique

les bruits initiaux (percussion, souffle, archet) et finaux (étouffoir, variation de hauteur) donnent des indications déterminantes sur l'instrument qui a joué. Les transitions entre les notes, notamment dans le jeu *legato*, donnent également des indications précises. Tout cela indique qu'il est très intéressant de se focaliser sur les transitoires pour identifier un instrument.

1.1.2 Les différentes approches en reconnaissance automatique d'instruments sur des phrases musicales

La Figure 1 montre le principe général d'une reconnaissance automatique d'instruments de musique. Une base de donnée d'instrument indexée à la main sert de base de donnée d'apprentissage. La distribution des paramètres pour chaque instrument (ou classe) peut ensuite être modélisée. Pour tenter de reconnaître un instrument inconnu, il faut donc comparer sa position dans l'espace des paramètres par rapport à celles des classes. La décision peut ensuite être prise.

La plupart des approches actuelles utilisent des calculs de paramètres sur des tranches de signal relativement courtes à l'échelle de l'audition[2] (typiquement 30 ms), comme le montre la Figure 2. Les approches diffèrent principalement par deux aspects : le choix des paramètres [2] et la méthode de décision [3]. Les paramètres utilisés peuvent être spectraux (centroïde

[2]L'identification d'un instrument par un sujet humain sur une si courte échelle est impossible

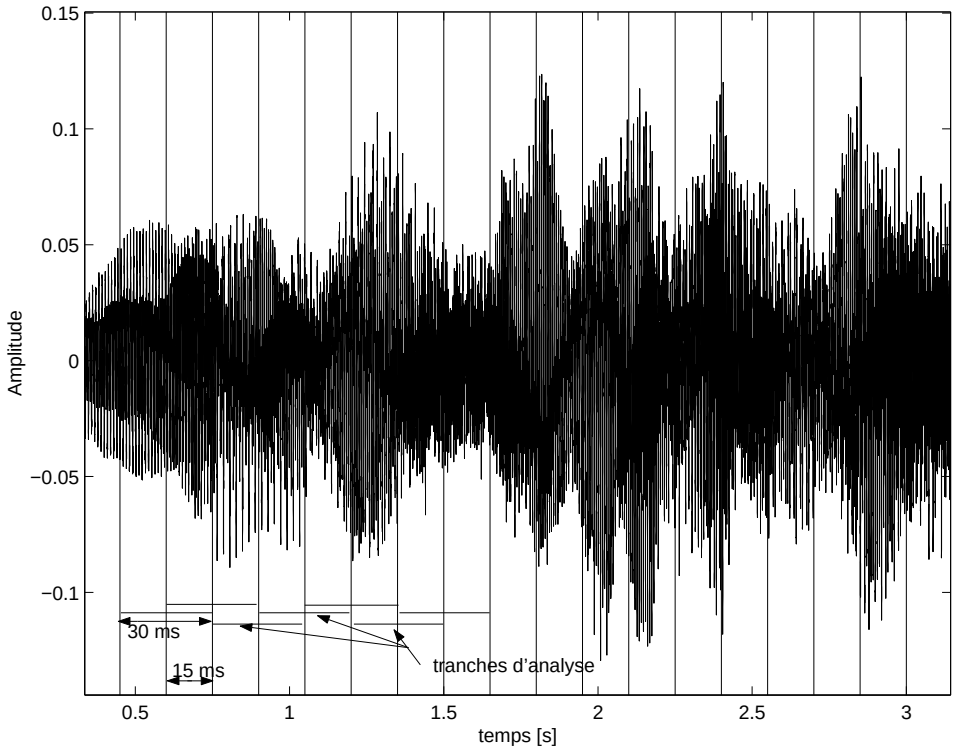

FIG. 2 – Réprésentation des tranches d'analyse sur un tracé temporel de signal musical

spectral, coefficients cepstraux ou MFCC[3] [4, 5, 6]), ou temporels (enveloppe d'amplitude, modulations de fréquence). Quant à la méthode de décision, elle s'appuie sur les *K plus proches voisins*, la classification bayésienne, l'analyse discriminante, les arbres binaires, les *Support Vector Machines* ou les réseaux de neurones.

Le système développé à l'ENST [4, 7] s'appuie sur un modèle de mélange de gaussiennes (GMM[4]). Selon cette hypothèse, la distribution des vecteurs à P dimensions[5] vérifie :

$$p(\mathbf{x}|\Omega_k) = \sum_{i=1}^{M} p_i^k b_i^k(\mathbf{x})$$

où la densité b_i^k est définie par :

$$b_i^k(\mathbf{x}) = \frac{1}{(2\pi)^{P/2}|\Sigma_i^k|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(\mathbf{x}-\mu_i^k)'(\Sigma_i^k)^{-1}(\mathbf{x}-\mu_i^k)\right)}$$

[3]Mel-scaled Frequency Cepstrum Coefficients
[4]Gaussian Mixture Model
[5]P = Nombre de paramètres calculés

Pour ce modèle, M est le nombre de gaussiennes, μ_i^k est le vecteur des moyennes et Σ_i^k la matrice de covariance. Les paramètres du modèle $\{p_i^k, \mu_i^k, \Sigma_i^k\}$ sont évalués par la méthode du maximum de vraisemblance, et la décision basée sur la méthode du maximum *a posteriori*.

1.1.3 Approche choisie

Nous avons vu que le calcul des paramètres s'effectuait sur toutes les trames du signal, indépendamment de leur contenu (excepté sur les trames de silence). On peut cependant noter que certains paramètres peuvent avoir une pertinence variable suivant que la tranche analysée appartienne à un transitoire (ou contienne un transitoire) ou à une partie tenue d'une note. En effet, on peut penser que les paramètres spectraux seront plus adéquats sur les parties tenues des notes du fait de leur relative stationnarité, tandis que des paramètres temporels seront plus intéressants pour la caractérisation des parties transitoires. Une paramétrisation adaptée pourrait donc s'appuyer sur une différenciation du jeu de paramètres suivant le type de la tranche. Une fois le signal segmenté, on pourra alors dans un premier temps effectuer une reconnaissance d'instruments uniquement sur les transitoires. Dans un deuxième temps, la décision pourra être prise en fusionnant les statistiques obtenues sur l'évaluation à partir des transitoires et celles sur les parties tenues.

Pour réaliser cette segmentation, il faut d'abord développer une fonction de détection de débuts de note (*onset*), qui permettra de segmenter temporellement les performances en notes ou accords distincts. Il s'agira ensuite, pour chacune de ces entités, de séparer les parties transitoires des parties tenues. Dans une première approche, nous ne tiendrons pas compte de la variabilité de la durée des transitoires selon les instruments (cf. 1.3). C'est une approximation très grossière qui devra certainement être affinée, par exemple en déterminant un critère de stationnarité dans l'optique d'un calcul adaptatif de la durée de transitoire. Nous négligerons également l'influence des transitoires d'extinction, qui ne sont pas détectés par nos méthodes de détection.

1.2 Cadre de l'étude

L'étude se concentrera sur des performances solo d'instruments classiques. Les tentatives de reconnaissance des instruments seront effectués sur dix instruments : saxophone alto, basson, clarinette, flûte, hautbois, cor anglais, trompette, violoncelle, violon, piano. Les signaux étudiés sont échantillonnés à une fréquence de $F_e = 32$ kHz.

Les tests de détection d'*onsets* seront étendus à quelques autres sources tels que des sons de synthèse, de guitare acoustique et électrique.

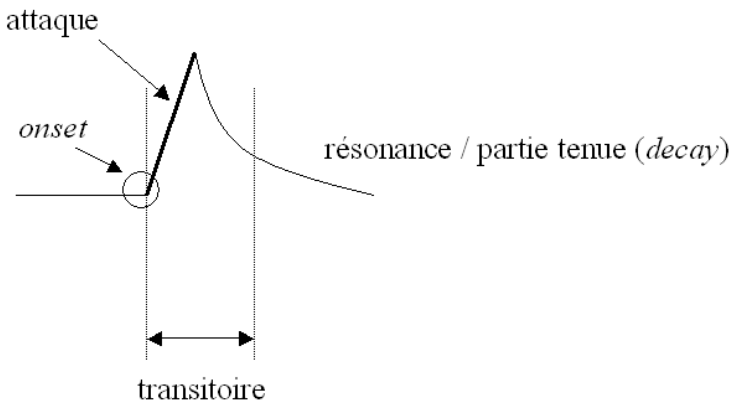

FIG. 3 – Transitoire, attaque et *onset*, d'après [1]

1.3 Les transitoires, les attaques et les *onsets*

Une distinction doit être faite entre transitoires, attaques et *onsets* [1].

1.3.1 Définition et caractérisation

La figure 3 représente la distinction entre ces trois concepts, telle que définie dans [1]. La *partie transitoire* d'une note est le temps pendant lequel le contenu spectral est rapidement variable. Il est suivi par la partie tenue pour les instruments à vent où à corde, ou la résonance pour les instruments à excitation percussive, comme le piano.

L'attaque fait partie du transitoire et définit la période pendant laquelle l'énergie augmente.

Quant à l'*onset*, il se définit par l'instant de départ d'une note jouée. C'est donc aussi la borne inférieure de l'attaque et du transitoire. Dans notre étude, nous chercherons à détecter les *onsets* grâce à des méthodes algorithmiques.

Pour un sujet humain, déterminer un *onset* pour une note isolée peut se faire facilement si l'on dispose de sa représentation graphique, ou à l'oreille avec une précision moindre. Comme nous le verrons par la suite, la précision de cette tâche diminue dans les phrases musicales, particulièrement dans certaines conditions d'enregistrement et pour certains instruments. On peut également s'intéresser à l'extraction de propriétés sur l'attaque et le transitoire. Par exemple, pour extraire la durée d'une attaque, une représentation de l'énergie du signal est nécessaire, la détermination de la durée à l'oreille n'est plus fiable. Enfin, définir la durée d'un transitoire est parfaitement arbitraire : la pluralité des variations spectrales empêche de définir un critère universel de "transitoirité". Cependant rien n'empêche de choisir un tel critère s'il permet d'obtenir une discrimination entre plusieurs instruments

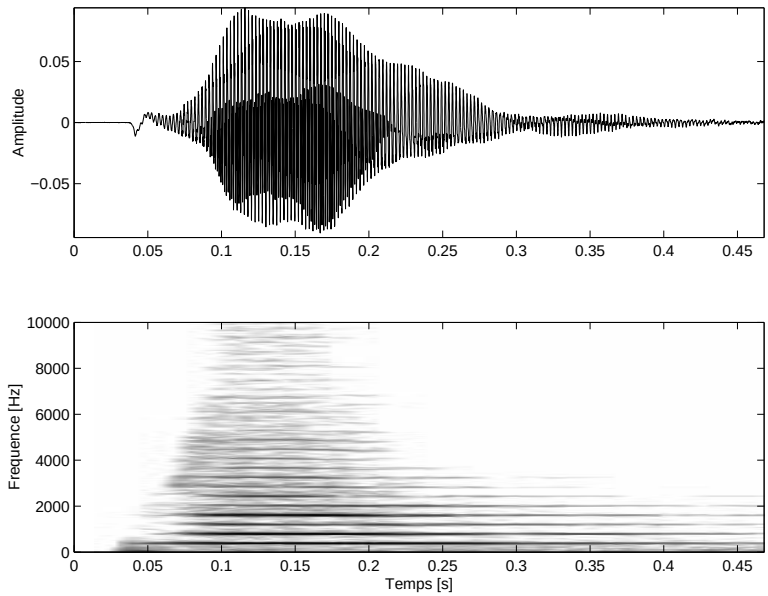

FIG. 4 – Tracé temporel (haut) et spectrogramme (bas) d'une note de flûte

lors d'une tentative de reconnaissance automatique.

1.3.2 Exemples de transitoires d'instruments

Les Figures 4, 5, 6 et 7 montrent respectivement des notes de flûte, de piano, de trompette et de violoncelle.

Le spectrogramme et l'enveloppe temporelle de la flûte (Figure 4) montrent une attaque relativement longue (80 ms), ainsi qu'une apparition d'harmoniques au cours de la note. On remarque également une grande quantité de bruit pendant tout la note.

Le piano (Figure 5) possède quant à lui une attaque très courte (de l'ordre de 15 ms), puis une diminution lente de son amplitude (ici pour une note tenue). La richesse spectrale est maximale peu après la fin de l'attaque.

La trompette (Figure 6) possède une attaque également courte (environ 25 ms), qui semble plus proche de celle du piano que celle de la flute sur le spectrogramme. On remarque cependant une explosion de bruit pendant l'attaque.

Enfin, le violoncelle (Figure 7) possède une attaque très longue (au delà de 500 ms) : le contenu spectral s'enrichit au cours de la note. Par contre le bruit de la corde frottée reste présent avec un niveau relativement constant.

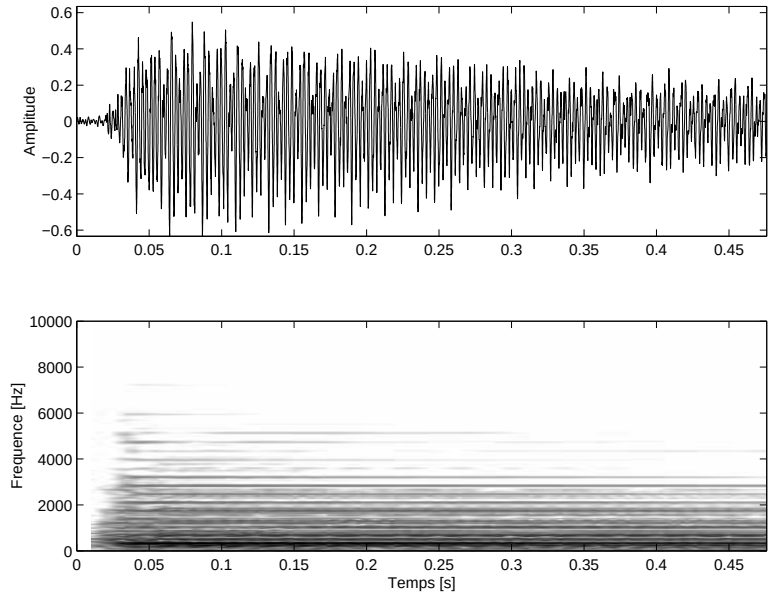

FIG. 5 – Tracé temporel (haut) et spectrogramme (bas) d'une note de piano

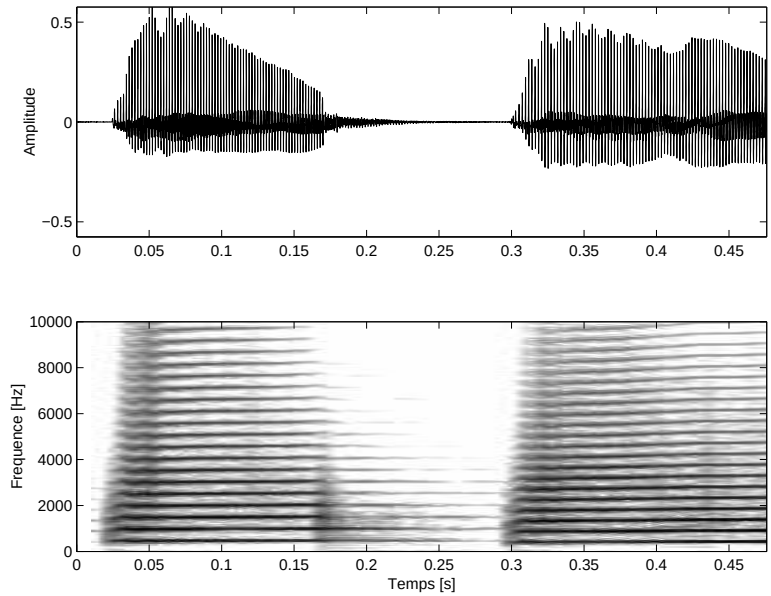

FIG. 6 – Tracé temporel (haut) et spectrogramme (bas) de deux notes de trompette

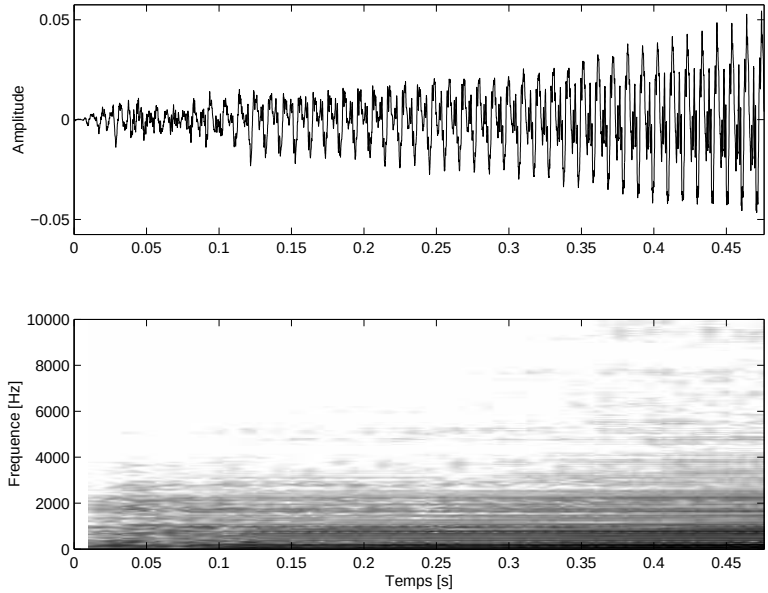

FIG. 7 – Tracé temporel (haut) et spectrogramme (bas) d'une note de violoncelle

1.3.3 Applications de l'extraction automatique d'*onsets* dans le cadre du stage

Reconnaissance des instruments : La variabilité de l'allure des transitoires a plusieurs conséquences sur l'application à la reconnaissance des instruments.

D'une part, la quantité d'information est considérable pour l'identification d'un instrument : le transitoire permet d'identifier le mécanisme de production d'un son, et ainsi donne une indication forte sur l'appartenance à une famille d'instruments. Pouvoir extraire de façon robuste la partie transitoire d'un son pourra certainement réduire la quantité d'information à traiter pour un résultat au moins équivalent à un découpage en tranche "à l'aveugle" du signal.

D'autre part, cette variabilité dans la forme et dans la durée pose justement un problème pour la robustesse de l'extraction de l'information. Tout d'abord, la détection automatique des temps d'*onset* semble difficile au regard des grandes différences entre les débuts de note, conséquence de la diversité des phénomènes physiques mis en jeu dans la production sonore : il faut trouver un critère suffisamment universel pour pouvoir détecter les *onsets* de tous les instruments. Ensuite, délimiter un transitoire semble ardu. Une analyse sur des fenêtre de durée adaptative au signal est nécessaire, mais bien entendu cette adaptativité ne peut pas se faire par rapport au type d'instrument, étant donné que cela constitue l'objectif du système.

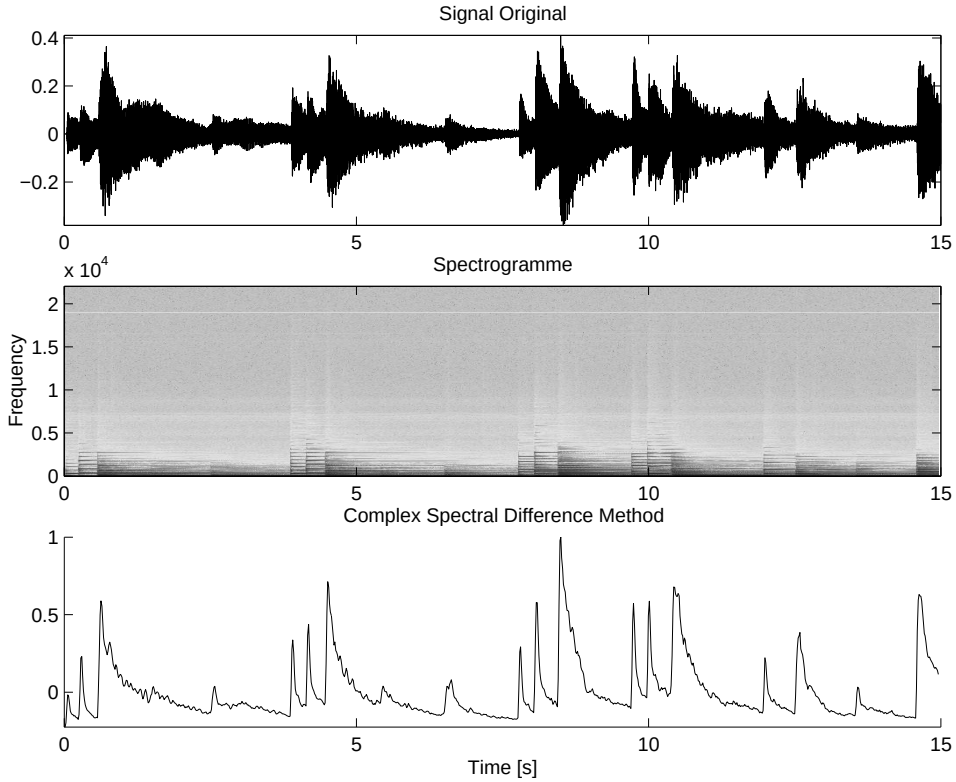

FIG. 8 – Exemple de fonction de détection. Haut : représentation temporelle, milieu : spectrogramme, bas : fonction de détection

Même si une information partielle sur la famille d'instrument pouvait être extraite à partir d'une analyse globale du signal, les différents modes de jeu (ex : *legato, staccato, pizzicato* pour le violon), auxquels correspondent des transitoires différents, empêcheront de décider d'un prototype de taille de fenêtre.

Extraction du rythme : La connaissance des *onsets* permet d'envisager l'extraction du tempo d'un morceau. Le travail de Miguel Alonso [8] est basé sur l'utilisation de fonctions de détections (cf. 1.4). Il pourra donc être intéressant de tester les algorithmes développés pour cette opération, même si les signaux étudiés sont très différents : la base de test pour l'extraction de tempo est constituée de mélanges complexes d'instruments.

1.4 Méthodes pour la détection de transitoires

Il existe différents types de méthodes pour détecter automatiquement les *onsets*. La majeure partie d'entre elles s'appuient sur le calcul de *fonctions de détection*, fortement sous-échantillonnées (exemple Figure 8). Ces fonc-

tions doivent présenter des pics prononcés pour les temps correspondant à des *onsets*, la fonction de détection idéale correspondant à un train d'impulsions de Dirac dont les abscisses sont situées à chacun de ces temps. Nous nous attarderons sur les méthodes les plus couramment utilisées et les plus performantes [1]. Le calcul de fonctions de détection se déroule généralement en trois étapes :

1. Le pré-traitement : il consiste à un traitement du signal avant réduction permettant d'accentuer les transitoires,
2. La réduction : c'est pendant cette étape qu'on utilise une ou plusieurs propriété(s) des transitoires qui les distingue des parties tenues afin d'obtenir des pics prononcés.
3. Le post-traitement : il permet d'accentuer les pics de la fonction de détection et/ou d'en enlever les pics supposés parasites.

1.4.1 Pré-traitements

Certains traitements facilitent la détection de transitoires. On peut notamment citer le filtrage en sous-bandes : la décision dépend alors de la combinaison des résultats des fonctions de détection sur chacune de ces sous-bandes. Un autre traitement possible met un jeu une suppression, ou à défaut une atténuation, des parties tonales du signal afin d'en accentuer les transitoires.

1.4.2 Réduction

Cette étape produit une fonction sous-échantillonnée présentant des pics marqués au niveau des *onsets*.

Méthodes temporelles : L'évolution temporelle des signaux musicaux montre un accroissement de l'amplitude au niveau des *onsets*. Une première idée peut donc être d'effectuer la détection sur l'enveloppe énergétique du signal. Cette approche peut fonctionner sur des signaux fortement percussifs, cependant elle atteint rapidement ses limites sur des sons entretenus (vents, cordes). Cette méthode est néanmoins utilisée pour la détection de rythme après décomposition du signal en sous-bandes et séparation transitoires/parties tenues.

Méthodes spectrales : Les méthodes spectrales ne nécessitent pas de pré-traitements : elles s'appuient sur l'incidence d'un *onset* sur le spectre. En posant la transformée de Fourier à court terme :

$$X_k(n) = \sum_{m=-N/2}^{N/2-1} c(nh+m)w(m)e^{-2j\pi mk/N}$$

où *w(m)* est une fenêtre de pondération, h l'espace entre deux centres de tranches d'analyse. On somme ensuite les modules des coefficients fréquentiels de cette fonction pour un instant n donné, en pondérant certaines fréquences par des facteurs $W(k)$, généralement dans le but d'accentuer les aigus. Une pondération particulière, $W(k) = |k|$, permet d'effectuer une telle opération (méthode *High Frequency Content*). Cette classe de fonctions de détection permet de repérer les transitoires dans la mesure où ils sont souvent considérés comme une courte tranche de bruit large bande, et fonctionne surtout sur des sons percussifs.

Une autre série de méthodes s'appuie sur les différences du spectre d'une fenêtre d'analyse à l'autre. On considère alors les spectres comme des vecteurs à N dimensions, on calcule ainsi la distance entre deux vecteurs successifs. Différents types de différences peuvent être choisies : on peut utiliser la norme L_1 ou L_2. On peut également utiliser la formule suivante, qui permet de travailler sur une différence redressée (méthode *Spectral Difference*) :

$$SD(n) = \sum_{k=N/2}^{N/2-1} \{H(|X_k(n)| - |X_k(n-1)|)\}^2$$

où $H(x) = (x + |x|)/2$, ce qui permet de ne tenir compte que des augmentations d'énergie.

Méthodes spectrales utilisant la phase : La phase peut aussi être prise en compte. Les transitoires sont caractérisés par une rupture dans l'évolution de la phase : on peut alors comparer la valeur de la phase réelle pour une fréquence donnée à son estimation dans l'hypothèse d'une sinusoïde à cette fréquence. On travaille alors sur la phase déroulée selon chaque fréquence, $\varphi_k(n)$, évaluée à partir de $X_k(n)$. Utiliser la phase déroulée permet d'éviter les sauts de phase qui ont lieu si on l'estime sur des intervalles de $[-\pi, \pi]$. Pour une sinusoïde stationnaire, la phase $\varphi_k(n)$ ainsi que celle de la fenêtre précédente $\varphi_k(n-1)$ sont utilisées pour calculer la fréquence instantanée entre les deux fenêtres d'analyse :

$$f_k(n) = \frac{\varphi_k(n) - \varphi_k(n-1)}{2\pi h} f_s$$

Comme la fréquence instantanée est supposée constante pour un signal stationnaire (c'est le cas des parties tenues des notes), on a :

$$\varphi_k(n) - \varphi_k(n-1) \simeq \varphi_k(n-1) - \varphi_k(n-2)$$

On définit alors la déviation de phase, caractéristique d'une rupture de la stationnarité d'un signal pour la bande de fréquence k :

$$\Delta\varphi_k(n) = \varphi_k(n) - 2\varphi_k(n-1) + \varphi_k(n-2)$$

On effectue ensuite une moyenne des déviations de phases absolues pour avoir une mesure globale de stationnarité (méthode *Phase Deviation*) :

$$\eta_p(n) = \frac{1}{N}\sum_{k=1}^{n} |\Delta\varphi_k(n)|$$

Cette méthode est plus efficace que les précédentes pour détecter les *onsets* des instruments percussifs, elle est cependant fortement sensible au bruit, étant donné qu'aucune attention n'est portée à l'énergie des bandes sur lesquelles on évalue la déviation de la phase.

On peut également combiner l'information sur la déviation de la phase et la différence spectrale. La différence mentionnée ci-dessus s'effectue alors dans un espace complexe : les éléments du vecteur sont $X_k(n)$ et non $|X_k(n)|$ (méthode *Complex Spectral Difference*). Il s'agit alors de définir l'estimation du vecteur à l'instant n dans l'hypothèse stationnaire en combinant les remarques précédentes :

$$\hat{X}_k(n) = |X_k(n-1)|e^{j(2\varphi(n-1)-\varphi(n-2))}$$

la différence s'écrit alors :

$$\Gamma_k(n) = \{|\hat{X}_k(n)|^2 + |X_k(n)|^2 - 2|\hat{X}_k(n)||X_k(n)|\cos(\Delta\varphi_k(n))\}^{\frac{1}{2}}$$

Comme précédemment, on somme selon les fréquences afin d'obtenir la fonction de détection :

$$\eta(n) = \sum_{k=1}^{n} \Gamma_k(n)$$

Méthodes Temps-échelle : On peut également utiliser des outils temps-échelle. Une transformée en ondelettes discrète revient à effectuer un filtrage par un banc de filtres. Les largeurs de bandes ont des tailles dyadiques : plus larges dans les hautes fréquences, plus étroites dans les basses. Cette méthode s'appuie sur la corrélation élevée entre les gros coefficients, typiquement lorsqu'un transitoire a lieu. On a alors des branches de gros coefficients dans l'arbre dyadique des coefficients.

On définit alors la fonction de détection comme suit :

$$\kappa_{s,p} = \sum_{(j,k)\in B[i]} 2^{js}|d_{j,k}|^p$$

où $d_{j,k}$ sont les fonctions d'ondelettes, $B[i]$ est la branche entière menant au coefficient petite échelle $d_{1,i}$. s et p sont deux paramètres permettant d'ajuster l'influence de certaines fréquences, souvent choisis égaux respectivement à 0 et 1. Le principal avantage de cette méthode est de fournir une grande précision en temps (de l'ordre de 2 échantillons). Cependant cette méthode reste sensible au bruit et la fonction de détection a tendance à suivre les oscillations basses fréquences, à moins qu'un pré-traitement ait été effectué (comme une séparation sinusoïdes/bruit).

Méthodes statistiques : Des méthodes statistiques ont été développées, et ont montré des résultats très convaincants, malgré une complexité importante. Cependant nous n'avons pas développé les algorithmes correspondants. Pour plus d'informations, on peut se référer à [1].

1.4.3 Post-traitements

Il s'agit maintenant de faire un traitement afin de faciliter l'extraction des pics. Certaines fonctions, comme celles dépendant de la phase, présentent un nombre de pics beaucoup trop élevé par rapport au nombre d'*onsets* dans le signal. On peut alors effectuer un filtrage passe-bas à l'aide d'une demi-fenêtre de Hanning. Prendre uniquement la moitié de la fenêtre permet de conserver les fronts montants des fonctions, car le maximum de la réponse impulsionnelle se situe sur le premier échantillon du filtre.

Un autre post-traitement consiste à retrancher la moyenne des fonctions de détection puis à les normaliser.

1.5 Extraction des *onsets*

Il s'agira d'effectuer une sélection des pics de la fonction de détection (*peak-picking*). Cela consiste à ne retenir que les pics susceptibles d'indiquer un transitoire. Si la fonction de détection est adéquate, les pics les plus grands indiqueront les transitoires, il reste donc à fixer un seuil au-delà duquel le pic sera considéré comme un transitoire.

L'algorithme de *peak-picking* le plus basique consiste donc à prendre les maxima au-dessus d'un seuil. Cependant, certaines fonctions de détection ont un niveau qui dépend de l'énergie instantanée du signal. Il est donc nécessaire d'utiliser un seuil adaptatif, qui dépend par exemple du niveau médian de la fonction de détection sur une fenêtre entourant le pic à traiter :

$$\tilde{\delta}(n) = \delta + \lambda \, median\{|d(n-M)|, ..., |d(n+M)|\}$$

Trois paramètres doivent être ajustés : le seuil statique δ, la pondération du niveau médian dans la fenêtre λ et la demi-taille de fenêtre M. λ a été fixé à 1.1 et la demi-taille de fenêtre M à 10 échantillons de la fonction de détection, c'est-à-dire environ 300 ms pour un son échantillonné à 44100 Hz. Ces valeurs permettent de ne pas tenir compte de pics qui seraient trop rapprochés. Quant au seuil statique, il permet d'avoir un certain contrôle sur le nombre de pics détectés : si le seuil est bas, la quantité de bonnes détections sera importante au prix d'un grand nombre de fausses alertes. Par contre un seuil haut ne permettra pas d'obtenir tous les *onsets*, mais le taux de fausses alarmes sera faible. Ces aspects sont exposés dans le prochain chapitre.

1.6 Évaluation d'une détection

L'évaluation des fonctions de détection se fait à partir de temps d'*onsets* référence. Bello [1] a choisi d'évaluer certaines fonctions de détection présentées ci-dessus sur une base de 1065 *onsets*. La fenêtre de tolérance prise autour du temps de référence est de 100 ms afin de pallier l'imprécision de l'annotation à la main.

Courbes *ROC* (*Receiver Operating Characteristic*) : Ces courbes permettent d'évaluer les performances relatives des fonctions de détection (cf. Figure 9). Chacune des fonctions de détection subit le même post-traitement. Comme le suggère le paragraphe précédent, le *peak-picking* se fera également avec des paramètres identiques et fixes pour toutes les fonctions, excepté le seuil statique qui permettra de parcourir l'espace défini par les axes bonnes détections et fausses alarmes.

Pour un ensemble d'*onsets* donné, on définit le taux de bonnes détections par le pourcentage d'*onsets* détectés par rapport au nombre total d'*onsets* à détecter. Quant au taux de fausses alarmes, il est égal au pourcentage de pics de la fonction qui ne correspondent pas à des onsets par rapport au nombre total de pics détectés. Le dénominateur dépend donc également du seuil statique qu'on fait varier. La fonction optimale serait donc une fonction possédant un point en (0, 100). On définit alors le seuil optimal comme le seuil pour lequel le point de la courbe est le plus proche de (0, 100).

Sur chacun des sous ensembles étudiés, on retrouve des résultats cohérents avec les prévisions :

- les méthodes spectrales sont relativement faibles pour les signaux non-percussifs (optimum à 81,7% bonnes détections (BD), 14,7% fausses alarmes (FA) pour HFC ; 87,1% BD, 8,6% FA pour SD)) tandis que celle basée sur la phase donne les meilleurs résultats (95,7% BD, 4.3% FA pour PD).
- les méthodes spectrales et basées sur la phase sont pratiquement équivalentes sur les signaux percussifs à hauteur déterminée (94,1% BD, 5,4% FA pour HFC)
- les méthodes spectrales sont plus efficaces sur les mélanges de sons complexes.

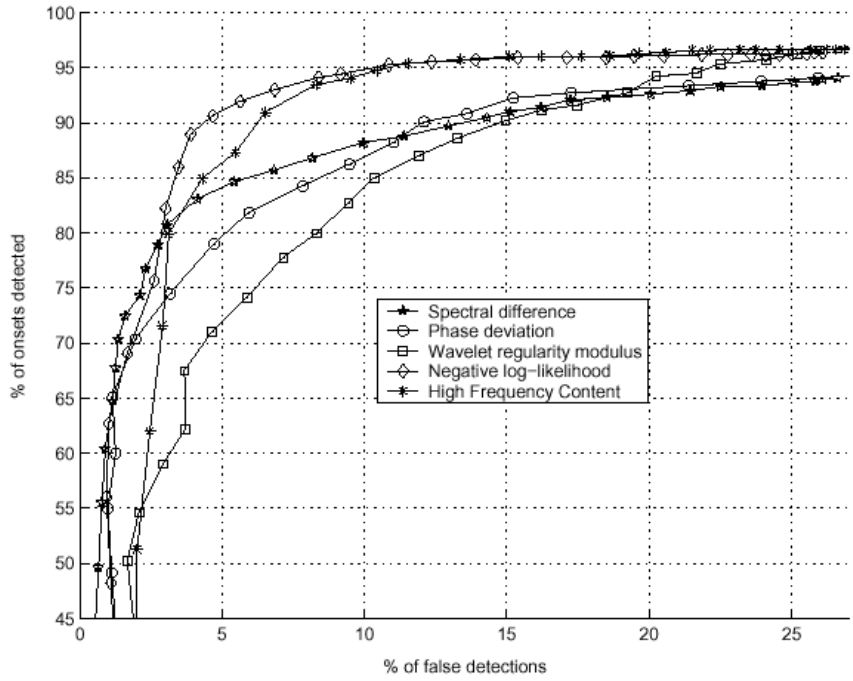

FIG. 9 – Courbe *ROC* obtenue par Bello

Deuxième partie
Travail effectué

2 Extraction des *onsets*

La première partie de mon travail a consisté à évaluer différentes fonctions de détection de transitoires sur une base d'instruments solos d'instruments. Les temps d'*onsets* de référence sont placés par des sujets humains. La question de la pertinence et de la précision de ces annotations a du être abordée avant de procéder à l'évaluation en tant que telle. Enfin, la méthode la plus adéquate à la segmentation pour la reconnaissance des instruments de musique sera choisie.

2.1 Méthodes développées

2.1.1 Fonctions de détection classiques

Les fonctions qui ont été développées sont les suivantes (cf. 1.4.2) :
- Méthode *High Frequency Coefficients*,
- Méthode *Spectral Difference*,
- Méthode *Complex Spectral Difference*,
- Méthode *Phase Deviation*.

Le développement des fonctions de détection est relativement trivial (cf. Annexe A). Les algorithmes correspondants prennent tous au moins trois paramètres : le signal à analyser, la taille de la fenêtre d'analyse, et les instants de calcul ou *hop size* (h). Ils retournent la fonction de détection (de période d'échantillonnage h/F_e, où F_e est la fréquence d'échantillonnage du signal), et le vecteur temps associé à la fonction. Toutes les fonctions de détection ont été inclues dans une interface de visualisation, qui permet également de choisir le post-traitement (filtrage passe-bas), le peak-picking et d'évaluer cette opération par rapport aux *onsets* relevés par les utilisateurs.

L'interface permet de choisir les fonctions de détection à visualiser, sur un fichier donné avec les labels d'un utilisateur donné. Les paramètres que l'utilisateur peut définir sont :
- Fenêtre d'analyse, *hop size*,
- Fréquence de coupure pour le filtrage passe-bas du post-traitement,
- Le seuil statique (δ), la pondération du niveau médian dans la fenêtre (λ) et la demi-taille de fenêtre (M) du *peak-picking*

D'autres fonctions de détection ont également été développées, en observant certaines propriétés des tracés des fonctions classiques. La Figure 10 présente l'application de certaines de ces fonctions de détection sur un enregistrement de guitare.

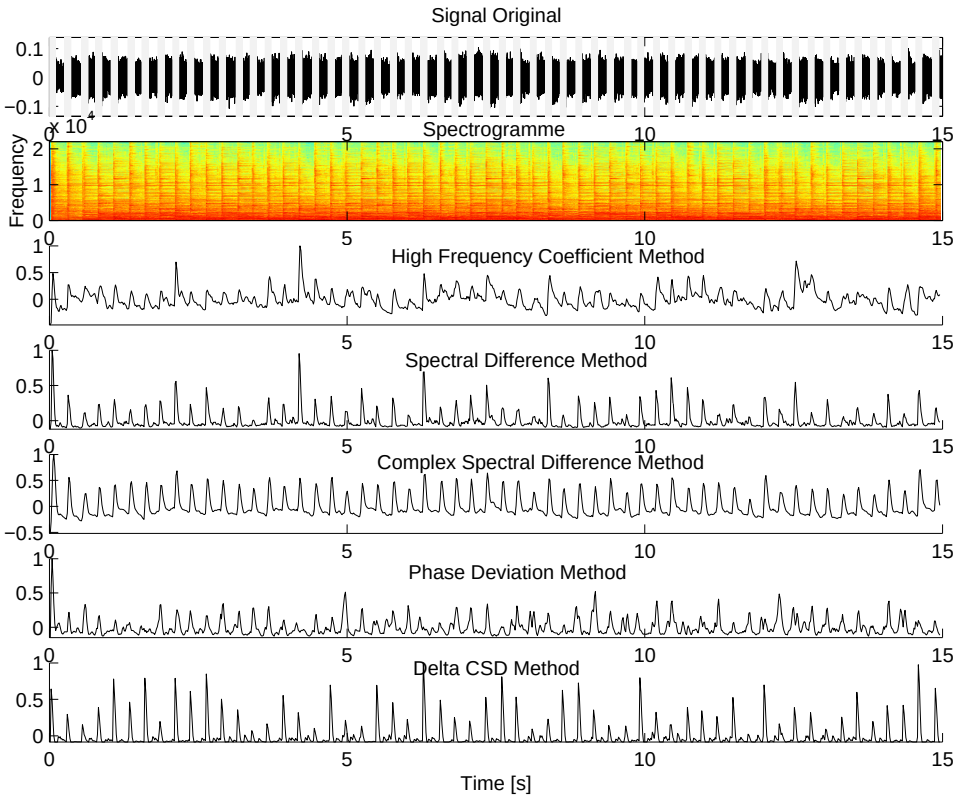

FIG. 10 – Fonctions de détection sur un enregistrement de guitare

2.1.2 Méthodes combinées

Certaines méthodes donnent une très bonne localisation des *onsets* en temps, mais donnent un trop grand nombre de pics. C'est le cas de la méthode *Phase deviation* : les variations de phases n'ont aucun rapport avec les amplitudes des signaux, cependant elles interviennent effectivement lorsqu'une nouvelle note est jouée. Cependant des variations de phase peuvent également intervenir de façon intempestive. Afin de déterminer si ces variations correspondent à un *onset*, il peut être intéressant de regarder si le pic de variation de phase est suivi d'une forte variation spectrale, caractéristique de l'attaque.

Le principal inconvénient de cette approche est l'augmentation du nombre de paramètres à régler : il faut tout d'abord déterminer ceux de la première fonction de détection, puis régler d'autres seuils pour le tri des pics. Ces méthodes n'ont pas été retenues pour les évaluations évoquées par la suite.

2.1.3 Autres fonctions de détection

Nous avons également développé d'autres fonctions de détection à partir des observations des résultats des fonctions classiques sur plusieurs fichiers. Il est apparu qu'une des fonctions (*Complex Spectral Difference*) donnait souvent un nombre de pics raisonnable au regard du nombre d'*onsets* relevés, mais que sa localisation en temps était peu précise. En effet, cette fonction, qui s'appuie à la fois sur les variations de la phase et sur celles du module du spectre, possède en général des pics relativement larges à cause de la prise en compte de ces deux variations qui ne sont pas toujours simultanées. Afin de rendre la localisation plus précise, on peut par exemple dériver le signal puis le redresser :

$$\Delta CSD = \max\{\Delta(CSD); 0\}$$

.

Ainsi, on détecte les maxima de pentes de la *Complex Spectral Difference*, qui s'avèrent plus proches des labels que les maxima de la fonction initiale lors des observations de tous les fichiers. Le redressement permet de ne pas tenir compte des pentes descendantes. On doit alors obtenir une proportion plus importante de bonnes détections, ce qui est confirmé par le tracé des courbes *ROC* (cf. 2.3).

2.2 Évaluation des fonctions de détection

2.2.1 Base de données

Afin d'évaluer correctement les fonctions de détection de la base, il est nécessaire de se doter d'une base d'*onsets* de référence fiable. Il s'agit donc

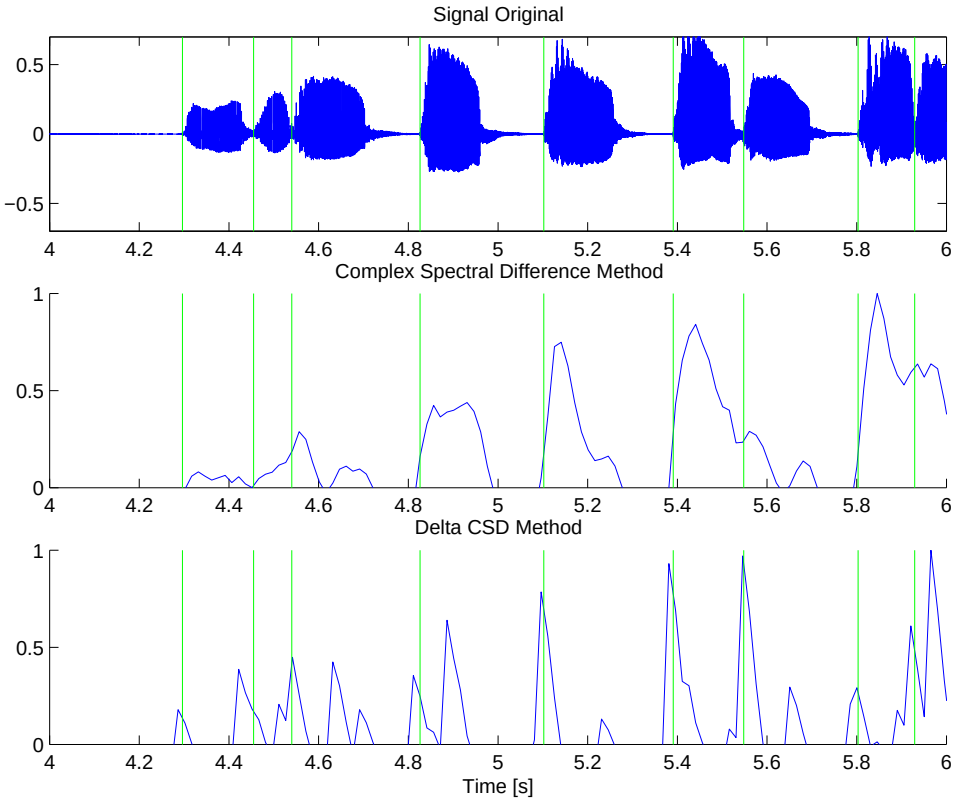

FIG. 11 – Représentation d'un enregistrement de trompette, évaluation des fonctions *Complex Spectral Difference* et *Delta Complex Spectral Difference*(haut : tracé temporel du signal, barres verticales indiquant les *onsets* de référence, milieu : *Complex Spectral Difference*, bas : *Delta Complex Spectral Difference*)

de déterminer les outils et les méthodologies adéquats pour parvenir à ce but. Cette question a fait l'objet d'un article soumis à la conférence ISMIR 2004 [9], disponible en Annexe.

Étant donnés les objectifs fixés dans cette étude, il s'agira de travailler sur une base de données de solos d'instruments. Voici les instruments étudiés :

- trompette,
- clarinette,
- saxophone,
- basse synthétique,
- violoncelle,
- violon,
- guitare saturée,
- guitare électro-acoustique,
- guitare jazz,
- piano.

Ces fichiers proviennent de deux sources : les trois premiers ont été enregistrés à l'ENST en chambre anéchoïque, les suivants ont été extraits de CD de la base de données RWC [10]. Les quatre premiers instruments sont purement monophoniques, les autres sont plus ou moins polyphoniques. Par exemple, le piano peut être considéré comme un instrument complètement polyphonique, tandis que le violoncelle ne l'est que partiellement : faire deux notes simultanément avec cet instrument est relativement peu fréquent. Cette base de donnée est relativement restreinte au regard de tous les instruments existant, cependant ils fournissent un éventail correct des types d'attaques pour une grande partie d'entre eux (instruments à anche, cuivres, cordes, attaques percussives).

2.2.2 Annotation des fichiers

Les *onsets* sont relevés par N sujets à l'aide d'un outil conçu à cet effet. Leur nombre pour un fichier est variable selon les sujets, et les temps relevés sont bien entendu différents. Les écarts entre les temps varient selon le type d'instrument : la moyenne des écarts est faible pour les instruments percussifs, plus élevés pour les instruments à son entretenu (à l'exception notable de la trompette qui possède une attaque nette).

L'outil d'annotation Nous avons développé une interface (cf. Annexe B) qui permet de visualiser le signal sonore, le spectrogramme ainsi que d'écouter le signal sur des fenêtres de taille désirée. Les fonctions de manipulations des labels lui fournissent un moyen simple d'ajuster précisément la position du label. La méthode utilisée par chacun des utilisateurs est la suivante :

1. Placement grossier du label à l'aide du spectrogramme,

2. Zoom sur une fenêtre d'environ une seconde autour du label,
3. Ajustement pas à pas à l'oreille (par pas de $5ms$).

Exploitation des annotations Il s'agit maintenant de ne garder que les labels qui font l'unanimité entre les utilisateurs et, à partir de leurs temps respectifs, de déterminer quel temps retenir pour la référence de l'évaluation. Afin de déterminer ces labels cohérents, on compare tout d'abord les annotations deux à deux, puis on ne retient les labels qui sont cohérents sur un cycle de comparaisons. Par exemple, si trois sujets ont participé à l'annotation, un label est cohérent pour tous les annoteurs s'il l'est dans les comparaisons entre les annoteurs 1 et 2, ainsi qu'entre 2 et 3 et enfin 3 et 1.

Pour une fenêtre de tolérance T_a donnée, si les deux labels sont dans la même fenêtre de tolérance, on considère qu'ils sont en accord. La fenêtre de tolérance T_a^{opt} est évaluée pour chaque fichier : elle correspond à 92% d'accord entre les différents annoteurs (cf. Figure 12). En effet, pour ce pourcentage, on atteint approximativement le pourcentage maximal d'accord entre les différents utilisateurs. Ce pourcentage correspond également à la limite de la zone dans laquelle l'évolution du nombre de labels cohérents est une fonction linéaire de T_a. Ce calcul doit être itéré pour chaque fichier, car le nombre de labels retenu pour chaque fichier en dépend. De plus, la connaissance de cette fenêtre de tolérance permettra d'évaluer les fonctions de détection en tenant compte d'une erreur d'annotation variable selon les fichiers.

Les labels retenus sont donc les moyennes des bons labels sur toutes les annotations. Comme les pics des fonctions de détection indiquant les *onsets* non consensuels sont quand même comptabilisés, on s'attend à avoir un taux de fausses alarmes assez élevé sur cette base d'*onsets* consensuels. L'alternative est de se baser sur le "meilleur" annoteur, du moins celui qui est le plus proche du consensus sur tous les labels. Cependant, se baser sur le meilleur annoteur n'a d'autre avantage que de réduire le taux de fausses alarmes : les résultats relatifs des méthodes, les unes par rapport aux autres, est le même.

Une fois les fenêtres de tolérance mesurées pour chaque fichier, on peut calculer les différences moyennes $\delta_{T_a^{opt}}$ entre les temps d'annotation des labels cohérents. On remarque que ces temps sont d'autant plus grands que les attaques sont molles (cf. Tableau 1). Naturellement, ils sont fortement corrélés aux fenêtres de tolérance T_a : le rapport $\rho = T_a^{opt}/\delta_{T_a^{opt}}$ est quasiment constant (entre 6 et 10).

2.3 Évaluation des fonctions - courbes *ROC*

Il s'agit maintenant de comparer les fonctions de détection à la base de données de labels ainsi constituée. Nous avons vu que la précision de l'annotation dépendait du type d'instrument, les fenêtres de tolérances T_{ROC} pour

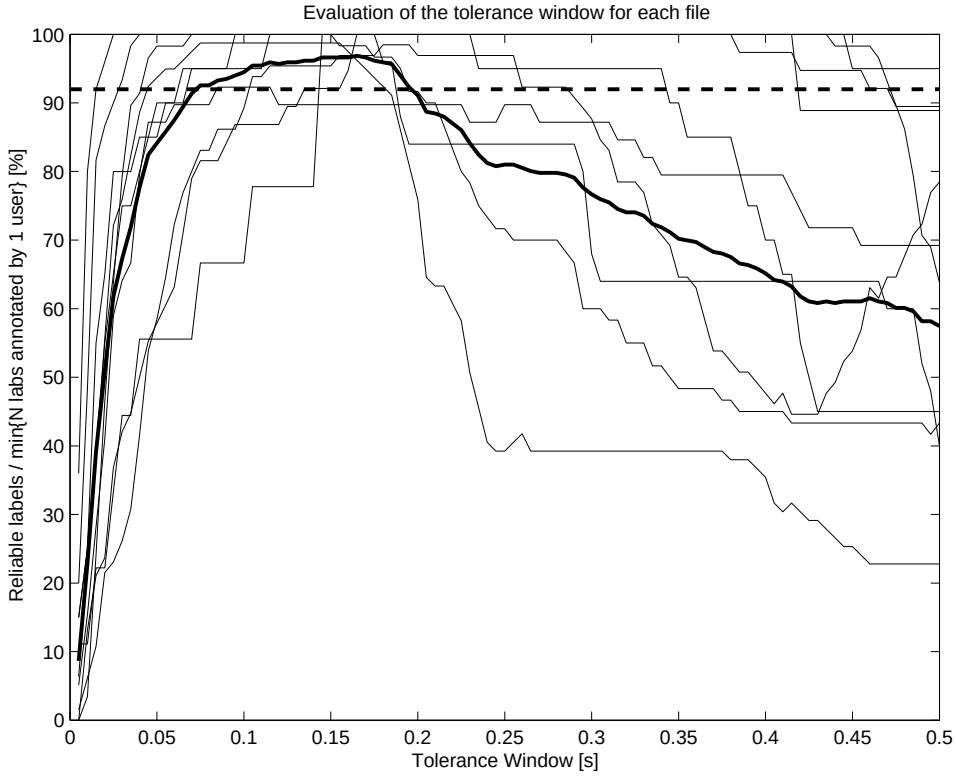

FIG. 12 – Évaluation de la fenêtre de tolérance en fonction du fichier (trait fin : courbe sur un fichier, trait épais : courbe sur tous les fichiers)

Fichier	Nombre d'*onsets* trouvés			Labels cohérents	T_a^{opt} [ms]	$\delta_{T_a^{opt}}$ [ms]	ρ
	An. 1	An. 2	An. 3				
clarinet	38	38	46	35	135	14,7	9,15
sax1	10	9	13	9	145	21,2	6,84
trumpet1	60	61	60	56	30	3,4	8,74
synthbass1	25	25	26	23	15	2,3	6,49
cello1	65	65	65	61	105	15,2	6,92
distguit1	20	22	21	19	65	7,3	8,89
guitar2	41	39	41	36	85	8,9	9,53
guitar3	58	58	58	56	45	7,3	6,12
violin2	79	79	79	73	45	6,4	7,01
piano1	20	20	20	19	70	6,9	10,04

TAB. 1 – Résultat de l'annotation : nombre de labels trouvés par chaque annoteur, nombre de labels cohérents, fenêtre de tolérance optimale T_a^{opt}, moyenne des écarts entre les annotations $\delta_{T_a^{opt}}$ et rapport $\rho = T_a^{opt}/\delta_{T_a^{opt}}$

l'évaluation des fonctions de détection seront donc dépendantes des écarts moyens $\delta_{T_a^{opt}}$ entre les annotations, lui-même dépendant de la fenêtre de tolérance T_a^{opt} pour laquelle le nombre d'*onsets* cohérents entre les annoteurs est maximal. Comme le rapport ρ est presque constant, nous effectuerons les évaluations en prenant T_a^{opt} pour fenêtre de tolérance pour l'évaluation ($T_{ROC} = T_a^{opt}$). Ceci est une différence notable avec les expériences d'évaluation effectuées précédemment, où la fenêtre de tolérance pour l'évaluation *ROC* est fixe (notamment dans [1]). Il semble en effet naturel d'être plus sévère sur l'évaluation des fonctions de détection sur des *onsets* percussifs que les annoteurs ont marqués consensuellement que sur les *onsets* non-percussifs. Le tracé des courbes *ROC* permet d'effectuer le compromis désiré entre le taux de bonnes détections et celui de fausses alarmes, en indiquant quel est le seuil statique à choisir pour la fonction de détection.

Sur les Figures 13 et 14 sont représentés les résultats des fonctions de détection développées pour deux types d'évaluation : la première correspond à une évaluation classique ($T_{ROC} = 100ms$), la seconde à une évaluation dépendante du fichier annoté ($T_{ROC} = T_a^{opt}$).

On constate donc que les performances des fonctions de détection est effectivement dépendante de la méthode employée pour les évaluer.

2.4 Choix de la fonction de détection

La fonction de détection doit être choisie en fonction de l'application : certaines fonctions de détection permettent de localiser précisément les *onsets*, d'autres sont avantageuses pour leur faible nombre de fausses alertes.

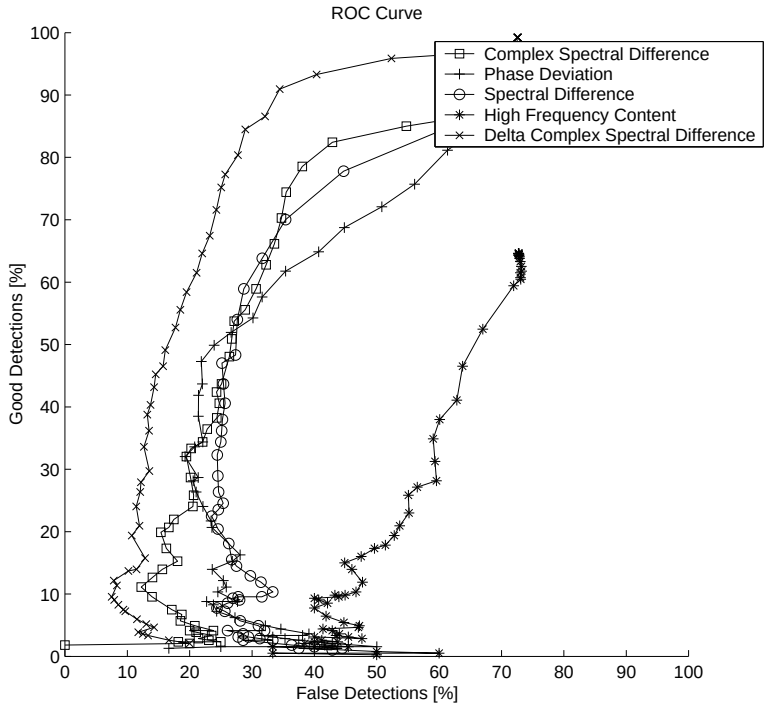

FIG. 13 – Courbes *ROC*, évaluation à fenêtre de tolérance fixe ($T_{ROC} = 100ms$)

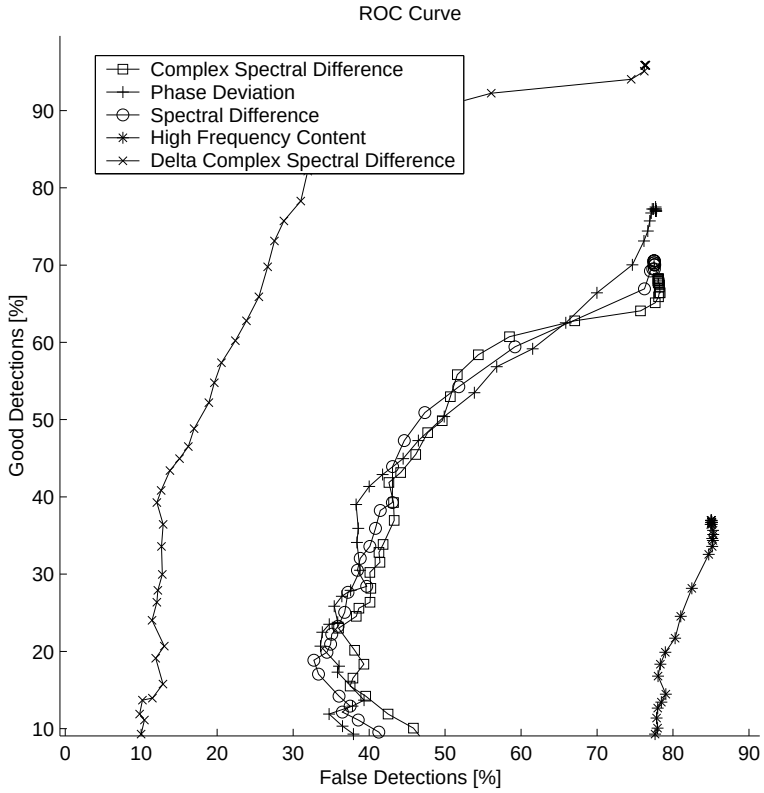

FIG. 14 – Courbes *ROC*, évaluation à fenêtre de tolérance dépendante du fichier ($T_{ROC} = T_a^{opt}$)

Dans le cadre de la reconnaissance d'instruments de musique, il s'agit de localiser précisément les *onsets*, avec un taux de fausses alarmes relativement bas afin de ne pas bruiter les distributions des instruments de musique dans l'espace des paramètres. Nous avons vu que la reconnaissance des instruments de musique demande une précision de 15 ms. Étant donnée la précision demandée pour cette application, le critère retenu serait donc :

$$T_{ROC} = max\{15ms, T_a\}$$

Cependant, il apparaît que les temps T_a ne sont jamais inférieurs à 15 ms (3 fois la précision de l'outil d'annotation, qui est de 5 ms). Cela signifie que pour un fichier donné, jamais les annoteurs n'ont été d'accord sur tous les temps d'annotation à 15 ms près, même sur les sons les plus percussifs. Par conséquent, on ne peut pas donner un crédit absolu aux annotations prises en référence : on ne peut pas savoir si les trames indiquées comme transitoire par la fonction de détection le sont effectivement. Nous retiendrons néanmoins la méthode qui donne les meilleurs résultats dans les conditions évoquées ci-dessus.

Sur la Figure 14, les courbes *ROC* obtenues pour les diverses méthodes développées sont représentées. Le choix se portera donc sur la fonction *Delta Complex Spectral Difference*. Pour l'implémentation finale d'indication des trames transitoires, on choisira le seuil pour lequel le point de la courbe est le plus près de l'optimum (point $(0, 100)$, cf. Figure 15).

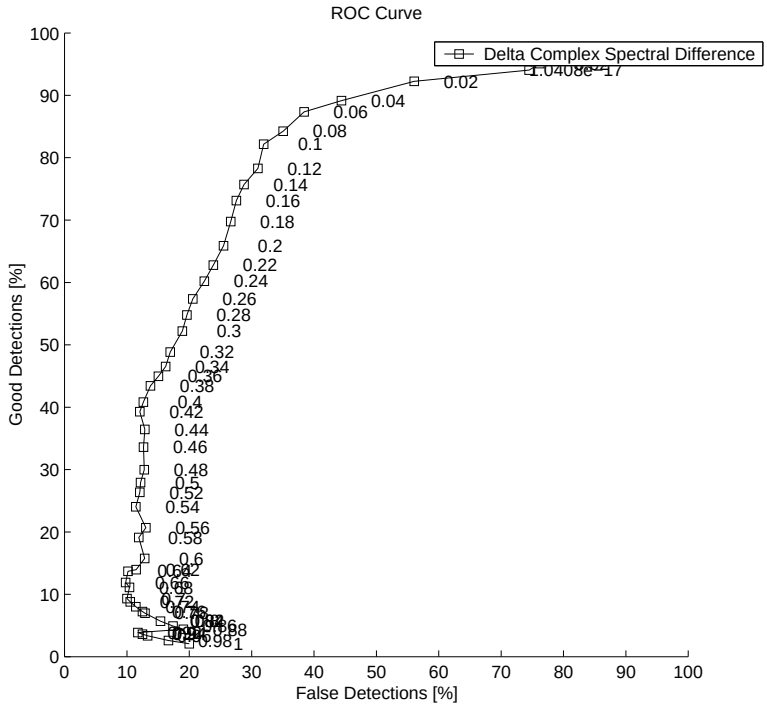

FIG. 15 – Courbe *ROC* de la fonction de détection choisie (Delta Complex Spectral Difference), seuils correspondant à chaque point

3 Choix de la paramétrisation des transitoires

3.1 Extraction des transitoires pour le système de reconnaissance des instruments

Le système développé à l'ENST [4] s'appuie sur un découpage des signaux en tranches (ou trames) de 30 ms, avec un recouvrement de 15 ms. La position des trames est donc sur une échelle fixée. La fonction de détection permet d'obtenir une détection des *onsets* à une précision de cet ordre pour les instruments percussifs. Elle est beaucoup plus difficile à évaluer pour les instruments à attaque molle étant donné que les références posées par les annoteurs possèdent une trop grande variabilité. On considérera néanmoins que les temps d'*onsets* indiquent la première trame d'une note. On peut très bien, dans une première approche, n'effectuer la reconnaissance que sur cette trame.

Cependant, nous avons vu que les transitoires peuvent s'étaler sur des durées plus longues. On peut donc inclure une ou plusieurs trame(s) suivant le premier pour obtenir le transitoire en entier. L'idéal serait d'avoir une durée de transitoire adaptative au signal, mais dans un premier temps, nous retiendrons une fenêtre de taille fixe (environ 50 ms, c'est à dire 2 trames).

3.2 Nouveaux paramètres

3.2.1 Principe - Intérêt de la transformée en ondelettes

Une fonction de détection d'*onsets* basée sur la transformée en ondelettes a été présentée en 1.4. L'avantage est de permettre d'obtenir une bonne localisation en temps. Or nous avons vu que la taille de fenêtre utilisée ($30ms$) ne permet pas d'être calé sur le début de la note, notamment pour les signaux percussifs dont on sait que les transitoires sont courts. L'idée est donc d'utiliser la transformée en ondelettes pour extraire une information plus localisée à l'intérieur d'une trame.

Une première idée peut être d'extraire des paramètres statistiques sur la fenêtre : on calcule le moment d'ordre 4 normalisé par le moment d'ordre 2 au carré :

$$M_4 = \frac{\sigma_4}{\sigma_2^2}$$

D'autres paramètres permettent d'extraire des informations beaucoup plus localisées : ils sont basés sur une branche de l'arbre des coefficients, par exemple celle correspondant au maximum de la fonction $\kappa_{s,p}$ (cf. 1.4), censée représenter le maximum de singularité [11].

Au regard des courbes représentant $\log_2(|d_{j,k}|)$ en fonction de j (log_2 de l'échelle), l'idée est d'introduire les paramètres suivant :

– la pente de l'asymptote (vers les petites échelles). Elle est évaluée par régression linéaire.

– le centroïde : $C_1 = \frac{\sum_{j=j_{\min}}^{J} j \log_2(|d_{j,k}|)}{\sum_{j=j_{\min}}^{J} \log_2(|d_{j,k}|)}$

– la courbure : $C_2 = \frac{\sum_{j=j_{\min}}^{J} j^2 \log_2(|d_{j,k}|)}{\sum_{j=j_{\min}}^{J} \log_2(|d_{j,k}|)}$

Le principal inconvénient de ces calculs de paramètre est d'intervenir sur une frame ne contenant pas forcément le maximum de $\kappa_{s,p}$ pour une note donnée. En effet, le maximum de cette fonction a tendance à suivre le maximum de l'attaque, si bien qu'il est souvent localisé en fin de trame d'analyse pour les instruments non percussifs.

On peut alors introduire d'autres paramètres similaires, mais basés sur la moyenne temporelle des coefficients sur une trame entière. Ces paramètres perdront alors leur principal intérêt *a priori* qui est la bonne localisation temporelle, mais la variabilité par instrument sera moindre. L'autre inconvénient de ces paramètres est d'être fortement similaires à des paramètres résultant de filtrage par octave, déjà utilisés dans le système de reconnaissance existant.

Une idée à développer est un pré-filtrage des composantes sinusoïdales du signal afin de les atténuer. Par ce biais, le maximum de $\kappa_{s,p}$ ne coïnciderait plus forcément au maximum d'énergie dans la trame.

3.2.2 Evaluation a priori des paramètres

Il est possible d'évaluer la pertinence des paramètres *a priori*, c'est-à-dire sans effectuer de reconnaissances des instruments. Il suffit pour cela de connaître les classes correspondant à chaque échantillon dont on évalue les paramètres. On définit alors le rapport IRMFSP introduit dans [12] et modifié dans [7] :

$$r = \frac{B}{T}$$

avec B, inertie inter-classe :

$$B = \sum_{k=1}^{K} \frac{N_k}{N} ||\mathbf{m}_{i,k} - \mathbf{m}_i||$$

et T, inertie totale :

$$T = \sum_{k=1}^{K} \left(\frac{1}{N_k} \sum_{n_k=1}^{N_k} ||\mathbf{f}_{i,n_k} - \mathbf{m}_i|| \right)$$

où N est le nombre total de vecteurs de paramètres évalués, N_k est le nombre de vecteurs de paramètres dans la classe k, $\mathbf{m}_{i,k}$ et $\mathbf{m}_i$ les moyennes respectives du vecteur de paramètres $\mathbf{f}_i$ sur les observation de la classe k et sur toutes les observation des K classes.

Plus le ratio r est élevé, plus le vecteur de paramètre associé discrimine les classes.

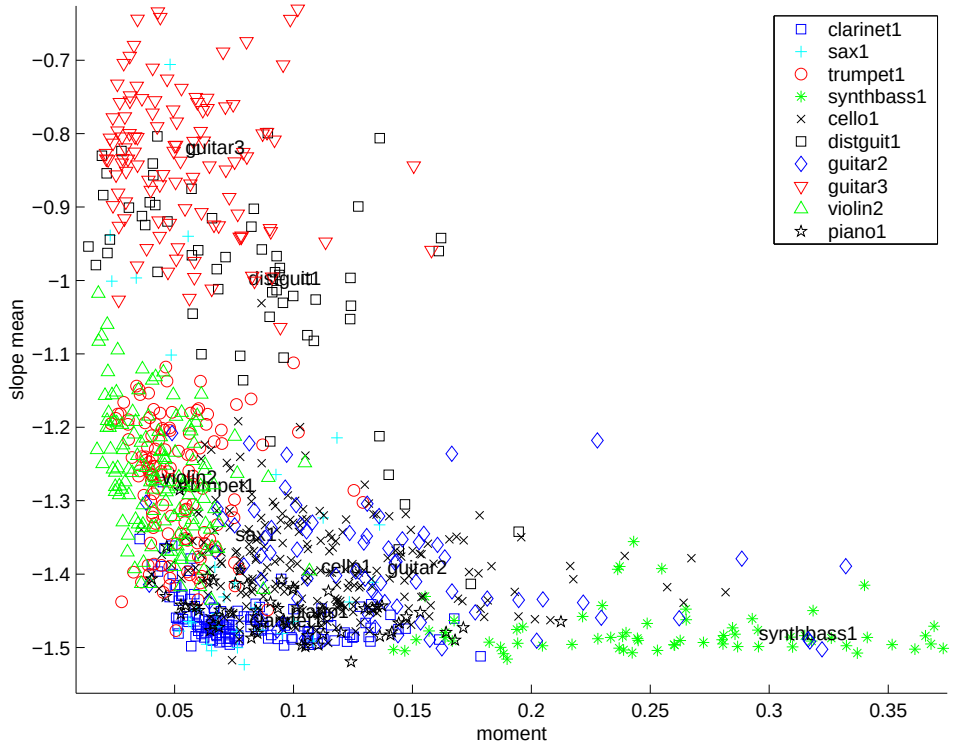

FIG. 16 – Répartition des échantillons des différents instruments

La Figure 16 montre la répartition des observations (calcul des paramètres sur les trames transitoires détectées automatiquement) dans le plan défini par les deux meilleurs paramètres au sens de l'IRMFSP (voir Tableau 2).

Malgré les réserves évoquées dans le paragraphe précédent, il apparaît que certains des paramètres calculés permettent une bonne séparation des classes. Cependant, les échantillons d'une même classe proviennent d'un même fichier, ce qui réduit la validité de la comparaison effectuée. La validité effective des paramètres testés pourra être évaluée correctement dans les tests de reconnaissance des instruments.

paramètre	Score IRMFSP
Pente moyenne sur la trame	0.081
Moment du 4^e ordre	0.075
Pente au max de κ	0.073
Centroïde au max de κ	0.068
Courbure au max de κ	0.066
Centroide moyen sur la trame	0.064
Courbure moyenne sur la trame	0.061

TAB. 2 – Scores IRMFSP pour les paramètres développés

4 Résultats sur les applications

4.1 Reconnaissance des instruments de musique

4.1.1 Évaluation de la reconnaissance d'instruments de musique sur les trames transitoires

On peut dans un premier temps tester la reconnaissance des instruments sur les trames transitoires uniquement, avec la paramétrisation de base (c'est-à-dire sans les nouveaux paramètres). Plus précisément, l'apprentissage se fait sur les trames indexées comme transitoires, les tests également. La conséquence néfaste de choix est de diminuer de façon drastique la quantité de données d'apprentissage : la proportion de trames transitoires par rapport aux nombre total de trames utiles est de l'ordre de 10%. Or la quantité de données d'apprentissage est primordiale pour effectuer une reconnaissance correcte. On peut également faire un apprentissage sur toutes les trames, et ne prendre les décisisions que sur les trames transitoires. Les décisions statistiques sont également effectuées sur peu de trames.

Instrument	Taux de Reconnaissance		
	AP toutes tr.	AP tr. trans.	AP toutes tr.
	DEC toutes tr.	DEC tr. trans.	DEC tr. trans.
Piano	99,07%	52,38%	54,76%
Saxophone	43,47%	28,73% *	31,80%
Basson	50,92%	33,58%	37,64%
Hautbois	75,94%	63,19%	65,77%
Clarinette	80,64%	46,60%	49,32%
Flute	76,16%	56,63%	58,61%
Cor anglais	71,00%	56,71%	53,25%
Violoncelle	94,26%	82,87%	82,51%
Violon	91,18%	80,45%	79,55%
Trompette	84,11%	77,51%	55,35%

Tab. 3 – Tests de reconnaissances (AP : apprentissage, DEC : décision, * : pour ce test, l'instrument testé n'arrive pas en première position dans les scores de reconnaissance)

Les résultats ne sont pas probants. Deux causes peuvent être envisagées : la diminution de la quantité de donnée utilisée pour l'apprentissage et les tests, et l'imprécision éventuelle de la fonction de détection pour certains types de signaux. En effet l'évaluation des fonctions de détection est relativement peu sévère pour les instruments à attaque molle (cf. 2.3). Par conséquent, les trames indexées comme transitoires peuvent très bien arriver trop tard par rapport au début réel de la note. On ne pourra cependant

évaluer correctement l'approche choisie que si on fusionne les décisions prises sur les parties transitoires et celles sur les parties tenues. On comparerait ainsi les résultats avec une évaluation classique, c'est-à-dire non différenciée en fonction de la tranche analysée.

4.1.2 Introduction des nouveaux paramètres

Le test n'a pas été effectué pour le moment.

4.2 Extraction de tempo

Ce sous-chapitre présente un résultat indépendant de la reconnaissance des instruments de musique, cependant il permet de valider les fonctions de détection développées.

4.2.1 Principe

L'extraction de tempo [8] est basée sur le calcul d'une fonction de détection. Celle-ci est seuillée dans le but d'enlever les pics non-significatifs grâce à un seuil dynamique du même type que celui défini en 1.5. A partir de la fonction de détection obtenue, on détermine le tempo par une extraction de périodicité. Les deux méthodes couramment utilisées sont le produit spectral et la fonction d'autocorrélation.

La première méthode est basée sur la multiplication du spectre par lui-même à des puissances entières :

$$S(e^{j2\pi f}) = \prod_{m=1}^{M} |P(e^{2\pi jmf})| \text{ pour } f < \frac{1}{2M}$$

Cela permet d'accentuer le pic de la fréquence fondamentale. Le maximum de cette fonction est ensuite extrait. On en déduit le tempo $\mathbb{T}$, avec $60 < \mathbb{T} < 200$.

La seconde méthode s'appuie sur la fonction d'autocorrélation non normalisée :

$$r(\tau) = \sum_{k} \tilde{p}(k+\tau)\tilde{p}(k)$$

L'espace entre les trois plus grands pics permet d'estimer le tempo.

4.2.2 Base d'évaluation

La base sur la quelle est évaluée la détection de tempo est constituée d'extraits sonores de divers styles, avec un grande variabilité dans leurs caractéristiques : 489 extraits musicaux répartis sur 10 styles. Les tempos de référence ont été annotés "à la main" : les annoteurs ont tapé le tempo sur un micro.

4.2.3 Résultats

Les méthodes ont été comparées à des algorithmes classiques d'extraction de tempo (cf. Tableau 4). La méthode Spectral Energy Flux a été développée par Miguel Alonso dans l'optique de réaliser la détection de tempo. Celle-ci est basée sur une différence spectrale, effectuée sur un nombre d'échantillon supérieur à deux pour un point fréquentiel k donné (ceci grâce à un différenciateur de Remez). Cependant, les résultats donnés par la méthode développée dans le cadre du stage (ΔCSD) sont tout à fait corrects en comparaison. Elle est en effet légèrement moins performante que la méthode *Spectral Energy Flux* pour la détection du rythme, par contre elle est plus fiable pour la détection d'*onsets* en général, notamment au niveau de leur localisation temporelle.

Méthode	Taux de reconnaissance
Paulus [13]	56,3%
Scheirer [14]	67,4%
CSD + SP.	57,3%
CSD + AC.	83,2%
ΔCSD + SP.	72,0%
ΔCSD + AC.	81,2%
SEF + SP.	84,0%
SEF + AC.	89,7%

Tab. 4 – Évaluation de la détection de tempo par différentes méthodes (CSD : Complex Spectral Difference, ΔCSD : Delta Complex Spectral Difference, SEF : Spectral Energy Flux, SP : Estimation par produit spectral, AC : Estimation par autocorrélation)

Conclusion

Résultats principaux

Dans cette étude, nous avons développé une méthode de détection d'*onsets* relativement performante dans les performances solo, en comparaison avec les méthodes classiques. Une méthodologie d'évaluation relative de toutes ces méthodes a été précisément définie : elle permet d'évaluer les fonctions de détection en tenant compte de l'erreur d'annotation humaine, variable selon les instruments. Nous avons ensuite introduit de nouveaux paramètres pour la reconnaissance d'instruments de musique, mieux adaptés aux transitoires. Leur pouvoir discriminant est intéressant, même si des recherches supplémentaires sont nécessaire pour optimiser leur pertinence.

Perspectives

La détection d'*onsets* n'est pas encore optimale : on devrait encore pouvoir se rapprocher de la courbe parfaite en introduisant des outils statistiques ainsi que des prédiction d'ordre supérieurs pour les calculs de différence spectrales et de déviation de phase, ce qui permettrait de ne pas tenir compte de variations temporelles sur des grandes échelles comme les vibratos et les trémolos. Les travaux de Miguel Alonso vont dans ce sens et sa méthode a prouvé une grande efficacité dans la détection précise du tempo.

Concernant l'évaluation des fonctions de détection, le logiciel Matlab développé (*Sound Onset Labelizer*) ainsi que la base de données d'*onsets* seront très prochainement mis en partage sur Internet afin de rendre cette méthode commune aux développeurs de méthodes de détection. Des annoteurs supplémentaires devraient aussi rendre notre annotation d'*onsets* plus robuste.

Enfin, la paramétrisation des transitoires pour les performances solo n'en est qu'à ses débuts. Les paramètres développés, très localisés pour certains d'entre eux, pourraient être plus significatifs s'ils étaient évalués uniquement dans la zone la plus "transitoire", ce que le découpage en tranche égales et les durées de transitoires fixées empêchent. Un approche plus adaptative s'impose. Le calcul du temps d'attaque serait également intéressant, à condition de l'extraire de façon robuste : c'est un paramètre qui possède des performances excellentes dans la reconnaissance automatique sur des notes isolées. D'autres voix sont également à explorer, notamment la prise en compte de l'évolution des paramètres au cours du temps en faisant intervenir des chaînes de Markov cachées (HMM) dans les modèles. Ces aspects seront étudiés prochainement avec Slim Essid.

Références

[1] J.P. Bello, L.Daudet, S. Abdallah, C. Duxbury, M. Davies, and M.B. Sandler. A tutorial on onset detection in music signals, to be published. *IEEE trans. on speech and audio process.*, 2004.

[2] A. Eronen. Comparison of features for musical instrument recognition. *Proc. of IEEE-WASPAA*, 2001.

[3] P. Herrera, X. Amatriai, E. Battle, and X. Serra. Towards instrument segmentation for music content description : a critical review of instrument classification techniques. *Reference missing.*

[4] S. Essid, G. Richard, and B. David. Musical instrument recognition on solo performance. *European Signal Processing Conference EUSIPCO*, 2004.

[5] A. Eronen and A. Klapuri. Musical instrument recognition using cepstral cofficients and temporal features. *Proceedings ICASSP*, 2000.

[6] J.C. Brown. Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. *Journal of the Acoustical Society of America*, 1999.

[7] S. Essid, G. Richard, and B. David. Musical instrument recognition based on class pariwise feature selection, submitted. *Proceedings of ISMIR*, 2004.

[8] M. Alonso, B. David, and G. Richard. Tempo and beat estimation of musical signals, submitted. *Proceedings of ISMIR*, 2004.

[9] P. Leveau, L. Daudet, and G. Richard. Methodology and tools for the evaluation of automatic onset detection algorithms in music, submitted. *Proceedings of ISMIR*, 2004.

[10] M. Goto. Rwc music database, published at http ://staff.aist.go.jp/m.goto/rwc-mdb/.

[11] S. Mallat. *Une exploration des signaux en ondelettes, pp. 174-179.* Les Editions de l'Ecole Polytechnique, 2000.

[12] G. Peeters. Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratio maximization. *115th AES convention, New York, USA*, 2003.

[13] J. Paulus and A. Klapuri. Measuring the similarity of rythmic patterns. *Proceedings of ISMIR*, 2002.

[14] E. D. Scheirer. Tempo and beat analysis of acoustic music signals. *J. Acoust. Soc. Am.*, 2004.

Troisième partie

Annexes

Annexe A : Codes

Voici les codes des fonctions :

1. *Spectral Difference* et *Complex Spectral Difference* (fonction **SpecDif**),
2. *High Frequency Content* (fonction **HFContent**),
3. *Phase Deviation* (fonction **PhiDev**),
4. *Delta Complex Spectral Difference* (fonction **DeltaSpecDif**).

```
------------------------------------------------------
function [SD, varargout] = SpecDif(Sigin, N, h, method,...
    varargin)

% SPECDIF
%   Cette fonction retourne la fonction de detection de
% transitoire par la methode Spectral Difference (et
% Complex Spectral Difference)
% ------------------------------------------------------
% Syntax:
% ------
%   [SD[, tSD]] = SpecDif(Sigin, N, h, method [, ...
% varargin])
%
% - SD: fonction de detection
% - Sigin : signal a etudier
% - N: taille de la fenetre
% - h: hop size
% - methode: * 'simple': technique basee sur la
% difference des modules
%            * 'complex': technique basee sur les
% differences des modules et des phases.

Sigin = Sigin(:);
L = floor((length(Sigin)-floor(N/2))/h);

w = hanning(N);

X = zeros(N,L);

Sigin = [zeros(floor(N/2)+1,1);Sigin]; % Decalage du signal
```

```
for n = 1:L
    X(:,n) = fft(w.* Sigin((1:N) + n * h));
end

SD = zeros(1, L);

switch lower(method)
case 'simple'
    % Methode Spectral Difference



    for n = 2:L
        SD(n) = 1/4 * sum( ((abs(X(:,n)) - abs(X(:,n-1))) +...
            abs( abs(X(:,n)) - abs(X(:,n-1)) )).^2 );

    end



    % Methode Complex Spectral Difference
case 'complex'
    EDist = zeros(N,1);

    CurrPhi = angle(X(:,1));

    SD(1) = 0;
    %UPhi = (unwrap(angle(X)));

    Phi = (unwrap(angle(X)));

    DevPhi = [zeros(N,2) (Phi(:,3:end) - 2*Phi(:,2:end-1) +...
            Phi(:,1:end-2))];

    for n = 2:L
        %CurrPhiEst = CurrPhi + 2 * pi * (1:N)' /N * h;
        %CurrPhi = UPhi(:,n);
        %DPhi = CurrPhi - CurrPhiEst;

        EDist = sqrt(abs(X(:,n-1)).^2 + abs(X(:,n)).^2 -...
            2*abs(X(:,n)).*abs(X(:,n-1)).*cos(DevPhi(:, n)));

        % Sommation decrite dans l'article
        SD(n) = sum(EDist);
```

```
    end


case 'improved'
    EDist = zeros(N,1);
    EDistPrec = zeros(N,1);



    SD(1) = 0;
    Phi = (unwrap(angle(X)));
    DevPhi = [zeros(N,2) (Phi(:,3:end) - 2*Phi(:,2:end-1) +...
            Phi(:,1:end-2))];

    for n = 2:L
%         CurrPhiEst = CurrPhi + 2 * pi * (1:N)' /N * h;
%         CurrPhi = UPhi(:,n);
%         DPhi = CurrPhi - CurrPhiEst;



        EDist = sqrt(abs(X(:,n-1)).^2 + abs(X(:,n)).^2...
        - 2*abs(X(:,n)).*abs(X(:,n-1)).*cos(DevPhi(:, n)));

        TrueEDist = EDist;

        % we remove the lowering coefficients to sharpen
        % the peaks:
        EDist = (EDistPrec < EDist).* EDist;

        % Sommation decrite dans l'article
        SD(n) = sum(EDist);

        EDistPrec = TrueEDist;

    end



otherwise
    error('option = Complex, Improved ou Simple');
end
```

```
%SD = 1/max(abs(SD)).*SD;

if ~isempty(varargin);
    Fs = varargin{1};
    varargout{1} = ((1:L)*h - (floor(N/2) + 1))/Fs;
end
------------------------------------------------------------
function [E, varargout] = HFContent(Sigin, N, h, varargin);

% HFCONTENT
%   Cette fonction retourne la fonction de detection de
% transitoire par la methode High Frequency Content
% ------------------------------------------------------------
% Syntax:
%
%     [E[, T]] = HFContent(Sigin, N, h[, Fs])
%
% - Sigin: Signal to analyze
% - N: Analysis window size
% - h: hop size
% - E: Detection function

Sigin = Sigin(:);

% ******************************
% Methode High Frequency Content
% ******************************


L = floor((length(Sigin)-floor(N/2))/h);

w = hanning(N);

X = zeros(N,L);

Sigin = [zeros(floor(N/2)+1,1);Sigin]; % Decalage du signal
% afin que les fenetres soient centrees sur les points
% d'evaluation

for n = 1:L
    X(:,n) = fft(w.* Sigin((1:N) + n * h));% STFT windowed
    %by W
end
```

```
WE = abs((floor(-N/2) : floor(N/2)-1))'; % weights for the
                                         %energy

E = zeros(1,L);

for n = 1:L
    E(n) = 1 / N * sum ((WE(:).*abs(X(:,n) .^2))');
end

%E = 1/max(abs(E)).*E;

if ~isempty(varargin);
    Fs = varargin{1};
    varargout{1} = ((1:L)*h - (floor(N/2) + 1))/Fs;
end

------------------------------------------------------------
function [nup, varargout] = PhiDev(Sigin, N, h, varargin)

% PHIDEV
%   Cette fonction retourne la fonction de detection de
% transitoire par la methode Phase Deviation.
% ------------------------------------------------------------
% Syntax:
%   [nup[,tnup]] = PhiDev(Sigin, N, h[, Fs])
%
% - nup: fonction de detection (vecteur ligne)
% - Sigin: signal mono a analyser
% - N: Taille de la fenetre d'analyse
% - h: hop size
% - Fs: sampling frequency

%Methode Phase Deviation

Sigin = Sigin(:);
L = floor((length(Sigin)-floor(N/2))/h);

w = hanning(N);

X = zeros(N,L);
% E = zeros(1,L);
```

```
Sigin = [zeros(floor(N/2)+1,1);Sigin]; % Decalage du signal
% pour que la fenetre soit centree sur le bin ou elle est
%evaluee.


for n = 1:L
    X(:,n) = fft(w.* Sigin((1:N) + n * h));% STFT windowed
    % by W
end

Phi = (unwrap(angle(X))); % unwrapped phase

DevPhi = [zeros(N,2) (Phi(:,3:end) - 2*Phi(:,2:end-1) +...
        Phi(:,1:end-2))];
nup = 1/N * sum(abs(DevPhi),1);



% %Modulation par l'energie du signal
% for n=1:L
%     E(n) = var(Sigin(h*n-N/2+1:h*n+N/2));
% end
%
% nup = E.*nup;

%nup = 1/max(abs(nup)).*nup;

if ~isempty(varargin);
    Fs = varargin{1};
    varargout{1} = ((1:L)*h - (floor(N/2) + 1))/Fs;
end

------------------------------------------------
function [DSD, varargout] = DeltaSpecDif(Sigin, N, h,...
 method, varargin)

% DELTASPECDIF
%   This function returns the derivate of the specified
% spectral difference function
% ------------------------------------------------------------
% Syntax:
% ------
%   [SD[, tSD]] = DeltaSpecDif(Sigin, N, h,...
```

```
% method [, Fs])
%
% - SD: detection function
% - Sigin : signal to analyze
% - N: window size
% - h: hop size
% - method: * 'simple': based on magnitude differences
%           * 'complex': based on complex differences.
% - Fs: sampling frequency

if ~isempty(varargin)
    Fs = varargin{1};
    if nargin == 6
        Nderiv = varargin{2};
    else
        Nderiv = 1;
    end
    [SD, tSD] = SpecDif(Sigin, N, h, method, Fs);
    varargout{1} = tSD;
else
    [SD] = SpecDif(Sigin, N, h, method);
end

DSD = SD;

for nderiv = 1 : Nderiv
    %DSD = PostTreat(DSD, 8, Fs, [1 0]);

    DSD = DSD(2:end) - DSD(1:end-1);

    DSD = [0 DSD]; %shift to obtain the same length.

    DSD = max([DSD;zeros(1,length(DSD))]); % rectification

end
```

Annexe B : Article présentant le logiciel *Sound Onset Labelizer* et la méthodologie associée

METHODOLOGY AND TOOLS FOR THE EVALUATION OF AUTOMATIC ONSET DETECTION ALGORITHMS IN MUSIC

Pierre LEVEAU, Laurent DAUDET
Laboratoire d'Acoustique Musicale
11, rue de Lourmel
75015 Paris - FRANCE
leveau,daudet@lam.jussieu.fr

Gaël RICHARD
GET - ENST (Télécom Paris)
46, rue Barrault
75634 Paris Cedex 13 - FRANCE
gael.richard@enst.fr

ABSTRACT

This paper addresses the problem of the performance evaluation of algorithms for the automatic detection of note onsets in music signals. Our experiments show that creating a database of reference files with reliable human-annotated onset times is a complex task, since its subjective part cannot be neglected. This work provides a methodology to construct such a database. With the use of a carefully designed software tool, called SOL (Sound Onset Labellizer), we can obtain a set of reference onset times that are cross-validated amongst different expert listeners. We show that the mean error of annotated times across test subjects is very much signal-dependent. This value can be used, when evaluating automatic labelling, as an indication of the relevant tolerance window. Finally, we illustrate the use of the reference database to compare several standard automatic onset detection schemes. The SOL annotation software is to be released freely for research purposes. Our test library, 17 short sequences containing about 750 onsets, comes from copyright-free music or from the public RWC database. The corresponding validated onset labels are also freely distributed, and are intended to form the starting point for the definition of a reliable benchmark.

1. INTRODUCTION

An increasing number of studies are concerned with the automatic extraction of note onset times directly from recorded audio, as this is useful in a wide range of signal processing applications : automatic transcription, adaptive audio effects, object-based coding, and more generally all information extraction techniques used for MIR (Music Information Retrieval). All these applications try to split the audio into segments that have homogeneous properties, e.g. spectral and / or statistical properties (see for example [1, 2, 3, 4, 5])

While this task is rather straightforward in the case of isolated notes, this can become a very difficult - and indeed ill-posed - problem for increasingly complicated sound files, from a single instrument melodic line to a full polyphonic orchestra. When many notes are played together, the notion of sound object may appear more relevant: for instance a chord can be considered as a single sound object. However, when this chord becomes broken (typically in a guitar slam) or when does it stop being a single object and start being a set of harmonically related notes ?

So far, the great majority of note onset detection schemes are based on the concept of "detection function" (DF). The DF is a highly sub-sampled version of the audio that exhibits peaks at the time instants where some properties change (e.g. energy, spectral content, etc ...) (see [6], [7] or [8] for a tutorial on onset detection). The performance of such schemes is usually evaluated through ROC curves (Receiver Operating Characteristics), a plot of the ratio of correct detections as a function of false alarms. The main problem arises from the definition of what a "correct detection" is, since it implies the existence of a reference that gives the time localization of "true onsets" with infinite precision. Unfortunately, such perfect reference does not exist, except in a very limited set of cases (e.g. synthesized music). Furthermore, one has to allow for the finite time resolution of the above-mentioned detection algorithms : a given onset candidate at time t is counted as correct if there exists a "true onset" within a time frame $[t - \tau, t + \tau]$. Finally, the performances of the different schemes proposed in the literature are not easily compared due to the lack of common database and protocol for their evaluation.

This paper hence addresses the two fundamental (but previously under-considered) following issues: how to construct a set of reference onset times, and what is a good choice for the time resolution τ. In most cases found in the literature, the set of reference onset times is given by human-annotated data. Here, we will keep the same fundamental assumption: the human perception is what we want as the ultimate judge; in other words *we would like our detection algorithm to give results that are as close as possible to what humans would do*. Amongst our findings, we have observed that, for a number of test files, this hu-

man annotation exhibits a significant dependency on the employed method, the underlying software, the listener himself, and above all on the type of music. This observation suggests that the reported performance of automatic onset detection schemes is at best over-simplified and at worst cannot be generalized (i.e. are only true with strictly the same experimental conditions). The main objective of this paper is a proposal for a common methodology and a common annotation tool, which in turn is used to build a common database of onset-annotated files. These tools and files are freely available in order to be shared by the widest community.

The paper is constructed as follows : after the definitions and a summary of mostly-used automatic detection techniques, different methods for onsets time annotation in audio files are described. Section 3 is devoted to the description of our Matlab-based annotation tool, the Sound Onset Labellizer. Section 4 describes our test database, and the differences in annotation results for three different listeners. The performances of three common onset detection schemes are given in section 5 on the test onset database. The concluding remarks (section 6) will focus on perspectives for the evolution of the software tool and the database.

2. ONSET CARACTERIZATION

2.1. Particularities of onsets in music signals

Before labelling the onsets in music signals, we must define what on onset precisely is. The commonly used definition is *the time when a note begins*. However such a definition does not remove all the ambiguities. First, all the studied music signals are recorded. That implies that the *real onset time*, when the player triggers the production of the note, is not necessarily visible/audible in the signal on which we work. Indeed this element depends for instance on the sensitivity of the microphone and the analog-to-digital conversion of the signal. However we will afterwards consider that an onset is the first detectable part of a note event in the recording *if the note were isolated*. Trying to interpolate the start of a signal when the recording does not contain the information is a too difficult task ; however, in the test set that we have chosen, we assume that the quality of the recordings is good enough to ignore this fact. Moreover, some unwanted or uncontrolled sound events may occur when music is recorded. For instance, the keys of the woodwind and the breathing of the player produce noises that we can hear if we pay attention to it, but they usually bear little aesthetic or musical meaning. Hence, when someone is asked to label onsets in a music signal, it is important to tell him if he must take into account these events.

As mentioned in introduction, picking out onsets when the notes are isolated is easy. Things begin to be more difficult when a musical sequence is played, e.g. in solo performances. For monophonic instruments, room effects are amongst the phenomena that disturb the decision, as the increased release time of a note can mask the onset of the following one. Polyphony adds other disturbances to our task: the broken chord can be considered as a sequence of notes or as a block. For bowed strings, it is also difficult to mark the onset of a note when the previous note is still played on another string. For mixed music, these difficulties are amplified. Even if the instruments are supposed to play together on a quantized temporal grid, most of the time the differences between the real onsets of the different instruments notes are not negligible, especially for slow tempi. All these elements suggest that onset detection is a relatively subjective task, and that the specifications on what we are looking for must be precisely expressed.

2.2. How to label an onset *by hand* (and by ear) ?

Hand-labeling onsets is a strenuous task, that takes time and requires extreme concentration. To label onsets in a music signal, a subject can principally use three methods:

- *signal plot*: this tool is very efficient to precisely and quickly label percussive signals. It can also be used as a secondary method: when an onset occurs, the wave shape can be altered.
- *spectrogram*: it can be used as a first approach. Because of the need to take large enough FFT windows to have a sufficient frequency resolution, this method is not very precise, but it helps to localize most onsets globally. Indeed, a common characterization of music onset is that they are generally accompanied by a burst at all frequencies.
- *listening to signal slices*: this method is the ultimate judge. Combined with visualizations, it allows an efficient labelling of signals; this is the most precise user-controlled method.

It is possible to imagine other representations of the signal, e.g. using wavelets, phase, or spectrogram scaled in bark. However, by sake of simplicity we have chosen to restrict the study to these three most commonly used methods ; and we have compiled these in a software tool called the *Sound Onset Labellizer* that is presented in next section.

3. HAND LABELLING

3.1. Annotation Tool: *Sound Onset Labellizer*

This tool has been developed to provide an easy-to-use and portable interface to the different labelling subjects. All annotators (or subject) have used the same software.

The screen of the GUI is divided in three parts: the upper one represents the spectrogram of the signal, the middle one its time domain waveform, and the lower the controls to manipulate sound files, labels and visualization windows (see Figure 1).

The spectrogram and waveform parts have the same time axis, and all the zoom operations act on both windows. The cursor on the right of the spectrogram allows a

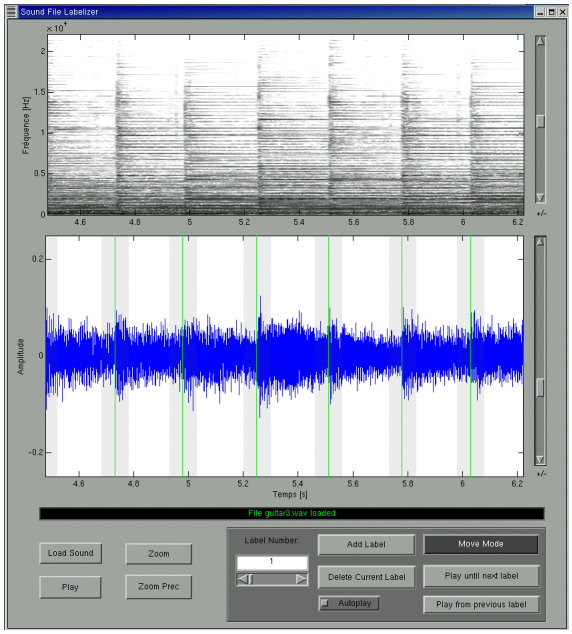

Figure 1. Interface of the *Sound Onset Labellizer*

setting of the contrast of the spectrogram, the one on the right of the signal waveform allows an amplitude magnification. The subject can play the sound visualized in the current window. The labels are put with a cursor, and can be moved by steps of 5ms to fit precisely the supposed onset time. Once a few labels are put, the subject can play the signal between two labels to evaluate if it contains only one note.

After a short learning time, all three annotators (the authors) have spontaneously adopted similar methods to label the onsets, following these steps:

1. *zoom* on a window containing a few notes (typically 1 or 2 seconds).

2. *label with a low precision* with the help of the spectrogram.

3. *precise adjustment* with the 'autoplay' option. It allows setting the label just before a new sound event occurs.

Note that no instruction nor guidance was given before the annotation except for the tool manipulation itself.

4. TESTS

4.1. Database contents

The labelling is evaluated on a first set of 17 sound files. Most of them have been extracted from the RWC database [9]. The sampling rate used throughout is 44.1 kHz.

The other ones comes from anechoic recordings made in the laboratory. The contents of our set are shared on a web site ([10]). The sounds extracted from the RWC database are not freely available, but the references of the files are indicated, as well as the start and end samples. The self-made recordings will be shared with a free access and all the rights of use for research purposes. The completion of our database is in progress and will be updated on the web site.

The set is composed of solo performances of monophonic instruments (e.g. trumpet, clarinet, saxophone, synthetic bass), polyphonic instruments (e.g. cello, violin, distorted and steel guitar, piano) and complex mixes in different music genres (e.g. rock, classical, pop, techno, jazz).

#	Content	Ref.	duration
1	Solo trumpet	ENST	14s
2	Solo clarinet	ENST	30s
3	Solo saxophone	ENST	12s
4	Solo synthetic bass	RWC	7s
5	Solo cello	RWC	14s
6	Solo violin	RWC	15s
7	Solo distorted guitar	RWC	6s
8	Solo steel guitar	RWC	15s
9	Solo electric guitar	RWC	15s
10	Solo piano	RWC	15s
11	techno	RWC	6s
12	rock	RWC	15s
13	jazz (octet)	RWC	14s
14	jazz (contrabass)	RWC	11s
15	classic 1	RWC	20s
16	classic 2	RWC	14s
17	pop1	RWC	15s

Table 1. Description of our database. The files with RWC reference are taken from the RWC database, those with ENST reference are recordings made in the lab and are available on our web site. Files are grouped in 3 categories: solo monophonic instruments, solo polyphonic instruments and complex mix. Full references of the RWC files can be found on the project's web site [10].

4.2. Evaluation Methods for the annotation

In a first step, we compare the annotation of two subjects. For each subject, the detected labels are counted on each file. Secondly, for each file, we calculate the time differences between corresponding labels where both subjects marked a given onset. The mean of these differences reveals the difficulty to annotate one file. Nevertheless, this second evaluation requires an arbitrary choice: we must decide to which extent two labels must be assigned to a same onset. We have set the maximum time difference between two corresponding labels at 0.1 second, considering that it represents an upper bound for the difference in both annotators' estimates of the same onset time. However, the optimal choice of this tolerance time needs further investigations.

File #	Number of labelled onsets			Number of consistent onsets	Average timing difference
	1	2	3		
1	60	61	60	60	3.9 ms
2	38	38	46	33	13.6 ms
3	10	9	13	6	11.9 ms
4	25	25	26	25	2.5 ms
5	65	65	65	58	14.4 ms
6	79	79	79	78	7.2 ms
7	20	22	21	20	8.9 ms
8	58	58	58	58	7.7 ms
9	41	39	41	37	9.9 ms
10	20	20	20	19	7.0 ms
11	56	56	56	56	4.7 ms
12	62	62	66	59	9.9 ms
13	56	52	56	47	11.7 ms
14	61	54	52	53	9.0 ms
15	49	49	53	38	15.8 ms
16	12	12	12	4	28.4 ms
17	32	40	41	27	11.7 ms
Total	744	741	765	678	10.5 ms

Table 2. Results of the hand-labelling process. Columns marked 1, 2 and 3 represent the number of onsets labelled by each of the test listeners. The next column indicates the number of consistent insets across listeners, used to construct our database of reliable onset times. The last column gives the mean timing error across listeners on the reliable onsets.

For each binary comparison, one of the annotated file is taken as reference. Then, for each of its labels, we look for a corresponding label in the second annotation within the defined tolerance window. When a corresponding label is found, it is marked as a *consistent label* for this given comparison.

To know the most reliable labels, we browse all the consistent labels of one comparison, and check that they are also consistent for the other comparisons. For instance, in our case where the annotation were conducted by three subjects, the consistent labels of the comparison between subjects 1 and 2 are selected and then it is checked that they are consistent in the comparison between subjects 2 and 3, and finally between subjects 3 and 1. By computing the average times of these labels between all the annotators, reliable onset times can be obtained. It is also possible to keep only the labels of the *best labeller* (the annotator whose labels times are the closest to these average label times).

4.3. Results

The number of labels set by each user for each file, the number of reliable labels and the mean of the differences between each annotations are shown in Table 2. We can first observe that the number of labels detected by the subjects is more variable when the number of notes playable at the same moment increases. An remarkable exception is techno music: the time is so quantized that all the listeners agree to the onset repartition. In the opposite case, mono instruments show poor performance when we can hear the breathing or the instrument keys. The extreme case is the “classic2” file (number 16), where the number of reliable onsets (4 out of 12) is so low that it is impossible to obtain statistically meaningful results with this file. This emphasizes the importance of the precise orders that have to be given to the listeners. A low number of reliable onsets is of course correlated with a high average difference between the labels time of each subject. This variable depends mostly of the percussiveness of the evaluated music signal.

5. APPLICATION

Three commonly-used onset detection algorithms are tested using the previous database. The chosen methods are the high frequency content, the spectral difference and the phase deviation (see [8] for a brief description). The first one is based on the sum of STFT magnitude across frequency bins with a linear frequency weighting. If it is the fastest method, it is also the less efficient. The spectral difference is based on the sum of the rectified spectral flux bins. The last one calculates the sum of phase deviations to detect breaks in strongly harmonic signals. These functions are then submitted to a peak-picking algorithm using thresholds based on the local median of the detection function ([11]). The static threshold can be chosen to obtain a compromise between high good detection and low false detection percentage.

The ROC curves which gives the good detection percentage as a function of the false detection percentage are given for the onset detection algorithm on Figure 2. This curves are obtained by varying the static threshold of the peak-picking algorithm. The ROC curves correspond to the average performance on our test database.

The false alarms are relatively high because of the low reliable onset number for all the subjects in comparison with the number of onset found by only one user, less under determined. However the curves corner are relatively sharp, meaning that lowering the threshold adds only false alarms after the optimal point. Finally, results are notably poorer then those reported in [8], but the database used here probably contains more difficult files.

6. CONCLUSION

In this paper, a fundamental aspect of the evaluation of automatic onset detection algorithms is studied. We have shown that the number of onsets detected by a listener is not only dependent on the music signal itself, but also on the guidance instructions given to annotators to mark the note onsets. This dependance suggests that onset detection algorithms could be evaluated with different tolerance

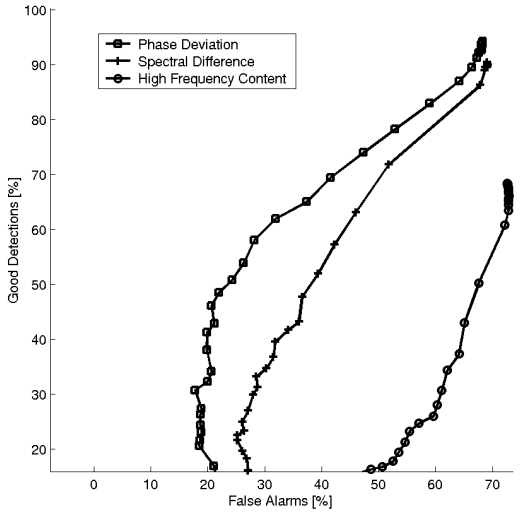

Figure 2. ROC curves for three onset detection algorithms : high frequency content (circles), spectral difference (plus) and phase deviation (squares).

windows on each type of file. For example, a 20 milliseconds tolerance window is appears to be acceptable for percussive signal, while it is definitely too short for music played by bow strings in order to take into account the differences of appreciation between annotators. However, this consideration must also be balanced in regards to the application of the automatic detection. For example, if tempo detection does not demand a very accurate onset localization, the estimation of the attack duration (for example for instrument recognition) would need a far more robust onset detection function.

Finally, in order to clearly contribute to future meaningful evaluation of onset detection algorithms, the test database, the software tool used to annotate the note onsets, an the set of reliable onset times are freely available for research purposes (except for the audio files extracted from the public database RWC for which only the position of the used audio segments are provided). The perspective of evolution of the database is to include more anechoic recording of solo performances, and to have more listeners annotating the database. An additional tool to visualize detection functions and comparisons between them is also in development in oder to have a standard evaluation software.

7. REFERENCES

[1] Klapuri, A. "Sound Onset Detection by Applying Psychoacoustic Knowledge", *Proceedings IEEE Int. Conf. Acoustics Speech and Sig. Proc. (ICASSP)*, pp. 3089–3092, Phoenix AR, USA March 1999.

[2] Laroche, J. "Estimating, Tempo, Swing and Beat Locations in Audio Recordings", *Proceedings IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 135–138, New Paltz, NY, USA October 2001.

[3] Davy M. and Godsill S., "Detection of abrupt spectral changes using Support Vector Machines: an application to audio signal segmentation", in Proc of the IEEE-ICASSP, Orlando, Florida, 2002

[4] M. Goto and Y. Muraoka, "A real-time beat tracking system for audio signals", Proc. of International Computer Music Conference, 1995.

[5] Rodet X. and Jaillet F. "Detection and modeling of fast attack transients" in Proc of IEEE-ICMC, 2001.

[6] Laroche, J. "Efficient Tempo and Beat Tracking in Audio Recordings", *J. Audio. Eng. Soc.*, vol. 51, No. 4, pp. 226–233, April 2003.

[7] Bello, J.P. Sandler, M. "Phase-based note onset detection for music signals" Proc. of IEEE workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), New Palts, USA, Oct. 2003.

[8] Bello, J.P, Daudet L., Abdallah S., Duxbury C., Davies M. and Sandler M., "A tutorial on onset detection in music signals", *to be published (IEEE trans. on ASSP)*.

[9] Goto M., RWC music database, published at http://staff.aist.go.jp/m.goto/RWC-MDB/

[10] Leveau P., Daudet L., G. Richard, "Database and tools for onset detection evaluation" to be accessible at http://www.enst.fr/˜grichard/ISMIR04/

[11] Kauppinen I., "Methods for detecting impulsive noise in speech and audio signals", in Proc. of DSP-2002, July 2002.