Formation de flux à partir d'une représentation objet de signaux musicaux polyphoniques

Renaud Houzet

27 août 2010

Stage effectué à l'Ircam encadré par Chunghsin Yeh, Mathieu Lagrange, Axel Roebel

Table des matières

I Introduction **5**

1 Problématique **5**

2 Termes, Notations **6**

3 Modèle sinusoïdal **6**

4 Modèle source-filtre, enveloppe, interpolation des partiels **7**

5 Modèle source-filtre et modèle acoustique **8**

II Modèles et etat de l'art **8**

6 Travaux en reconnaissance d'instrument **8**
6.1 Travaux de Eronen et Al. 9
6.2 Travaux de Kitahara et Al. 9
6.3 Conclusion sur la classification d'instrument 10

7 Travaux de Heittola et Al. **10**

8 Travaux de Miyamoto et Al. **11**

9 Travaux de Burred, modèle temporel simple **11**
9.1 Essais expérimentaux 12

10 Travaux de Graham Grindlay et Dan Ellis **14**

11 Conclusion **14**

III Elaboration d'une mesure de similarité de timbre pour les sons quasi-harmoniques 15

12 Présentation de la base de donnée utilisée **15**

13 Observation empirique des données brutes **15**

14 Mesure de similarité de timbre entre deux sons quasi-stationnaires **16**
14.1 Discrétisation de l'échelle des fréquences 16
14.2 Autres mesures de similarité entre enveloppes 17
14.2.1 MFCC 17
14.2.2 MFC-log, MFC-lin et MFC-exp 18
14.2.3 Entropie relative, divergence Beta, Itakura Saito, et Kullback Leibler 18

15 Présentation de quelques algorithmes utilisés **18**
15.1 Principal Component Analysis 18
15.2 Méthodes à Noyau 18
15.3 K-means 19
15.4 Kernel K-means 19
15.5 Normalized Cut 19
15.6 K-means déformé 20

16 Det Curve comme indice de qualité d'une mesure de similarité **21**

17 Détermination empirique des paramètres de fonction de compression **22**
17.1 Mesures de performance d'un algorithme de clustering 23

IV Algorithmes proposés **27**

18 Algorithmes déformés **27**
18.1 Factorisation de matrice avec pondération . . . 27
18.2 Application à notre problème de clustering timbral et à l'estimation de l'enveloppe . . . 27
18.3 comparaison des algorithmes pour l'estimation d'enveloppe . . . 28
18.4 Résultats pour le problème de clustering timbral . . . 30

19 Algorithmes K-means dans l'espace source-filtre **31**
19.1 Problème de la régularisation . . . 31
19.1.1 régularisation de Tikhonov . . . 31
19.1.2 à l'aide d'un programme quadratique . . . 32
19.2 Interpolation matricielle des partiels . . . 32
19.3 K-means dans l'espace des enveloppes . . . 32
19.3.1 Pourquoi les centroïdes devraient être des enveloppes . . . 32
19.3.2 algorithme K-means dans l'espace des enveloppes . . . 33
19.3.3 Réduction de la dimension des vecteurs d'enveloppe . . . 34
19.4 K-means avec modèle de source multiplicative . . . 35
19.5 K-means avec modèle de source additive . . . 35
19.6 Autre linéarisation du modèle . . . 36
19.7 Le problème de la normalisation de l'amplitude . . . 37
19.8 L'évolution des paramètres de régularisation . . . 37
19.9 Résultats pour notre problème de clustering . . . 37

20 Pondération optimale des Features dans l'espace de description **38**
20.1 critère proposés . . . 39
20.1.1 premier critère . . . 39
20.1.2 second critère (Fisher criterion) . . . 39
20.1.3 troisième critère (critère quotient) . . . 39
20.2 écriture du gradient . . . 40
20.3 minimum locaux . . . 40
20.4 optimisation pour Kernel K-means ou pour Normalized cut . . . 40
20.5 optimisation du critère pour Normalized Cut . . . 41
20.6 optimisation pour kernel K-means . . . 42

21 Estimation des paramètres du modèle de timbre en polyphonique **43**
21.1 méthode fréquentielle pour résoudre l'overlap . . . 43
21.2 méthode temporelle proposée . . . 44

V Conclusion et Bibliographie **46**

VI Annexe **48**

22 K-means **48**
22.1 dans l'espace de description . . . 48
22.2 Kernel K-means . . . 49

23 Algorithmes sur les graphes **49**
23.1 Normalized Cut . . . 49
23.1.1 cas à 2 classes . . . 49
23.1.2 cas à K classes . . . 50
23.1.3 Spectral Clustering . . . 51
23.1.4 équivalence entre Normalized Cut et weighted Kernel K-means . . . 51
23.2 un algorithme multi-niveau sans calcul de vecteurs propres . . . 52
23.3 Conclusion . . . 52

24 Algorithme K-means déformé **52**

25 PCA déformé **53**

26 NMF déformé **54**

27 Semi-NMF déformé **54**

28 Fischer / Linear Discriminant Analysis **55**

29 Principal Component Analysis **55**
29.1 suppositions probabilistes 55
29.2 démonstration empirique de l'optimalité de la transformation PCA 56
29.2.1 autre formulation du problème PCA 56
29.3 Les centroïdes obtenus par clustering K-means sont dans l'espace PCA 57

30 Non-negative Matrix Factorization **57**

31 Kernel Principal Component Analysis **57**
31.1 Formulation explicite des points dans l'espace de redescription (Empirical Feature Space) 58

32 MFCC et ondelettes **58**

33 Generalized EM pour réaliser un clustering en classes d'enveloppes avec modèle de source multiplicatif **59**

34 Generalized EM pour le modèle de source multiplicative **60**

35 EM pour le modèle de source additive **61**

Première partie
Introduction

1 Problématique

Dans le cadre de l'extraction de fréquences fondamentales multiples, l'information de timbre consistant entre autre en l'évolution des partiels est encore peu utilisée. Celle-ci pourait permettre d'une part de classer ou segmenter le flux de notes extrait en diverses classes d'instruments ou sources sonores, ceci pour une meilleure représentation idéalement sous forme d'une partition à plusieurs portées. D'autre part, un modèle de timbre pour chaque source élaboré dynamiquement ou connu a priori aiderait à résoudre des cas difficiles comme les partiels superposés ou la séparation des parties d'attaques. Par source nous entendons ici, de manière volontairement ambigue, soit un ensemble de trajectoires dans le temps plan-fréquence regroupées perceptivement comme une seule et même note, soit comme une classe dans laquelle il est plausible de regrouper plusieurs notes ayant plusieurs propriétés communes notamment en terme de timbre.

Dans ce rapport, le nombre d'instrument est supposé peu élevé ($5-6$ instruments au maximum) et chaque instrument est supposé jouer un nombre non négligeable de notes, avec un ensemble restreint de modes de jeux.

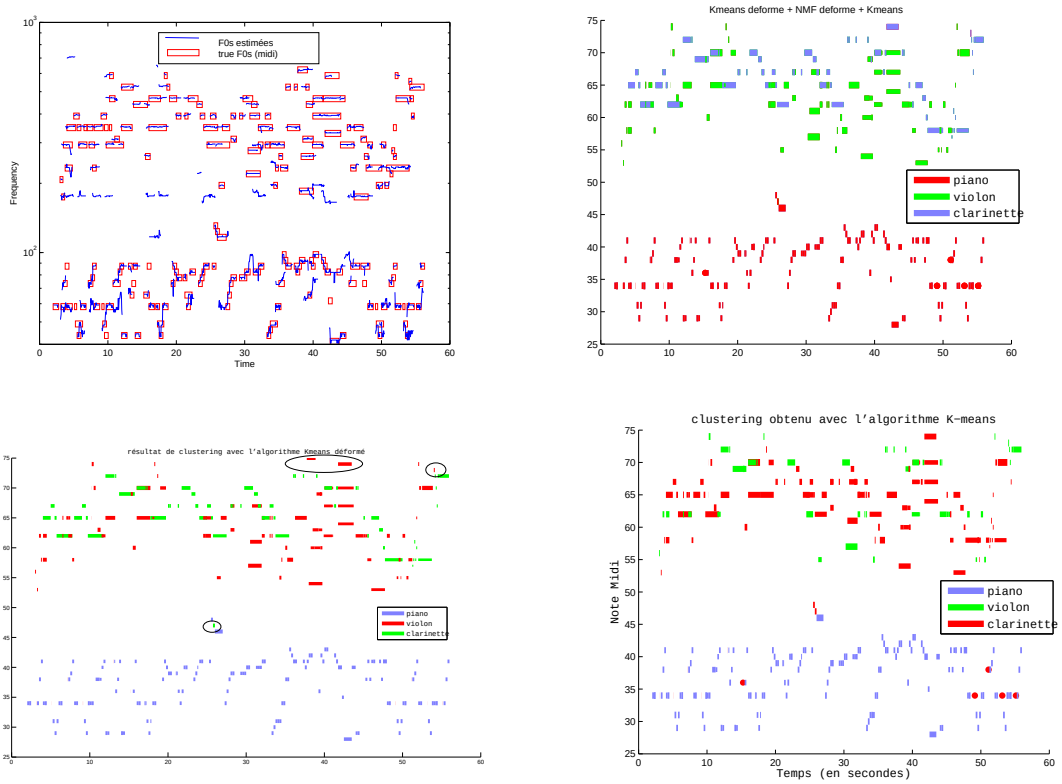

FIGURE 1 – en haut à gauche : F0s obtenues par l'algorithme multipitch développé à l'Ircam. en haut à droite : partition midi du morceau (Jazz 42 de la base RWC), réarrangé pour piano violon et clarinette, colorée en fonction de l'instrument. en bas à gauche : Clustering obtenu par algorithme K-means déformé (6% d'erreur). en bas à droite : Clustering obtenu par algorithme K-means (36% d'erreur).

Le problème posé dans ce stage est le suivant : à partir d'un signal audio polyphonique, est-il possible, en s'aidant notamment de la sortie d'un algorithme d'analyse multipitch (analyse de fréquence fondamentale multiple [4, Yeh2010], [23, Klapuri2008]) , de retrouver les divers instruments / sources présentes (figure 1). A l'heure actuelle, les algorithmes multipitch donnent le plus souvent en sortie, pour chaque trame, un ensemble de fréquence fondamentales (F0s)détectées. Nous utilisons donc ces F0s pour estimer l'amplitude des partiels de chacune des notes détectées que nous utilisons ensuite pour réaliser un clustering [1]. Une première question se pose quant à la faisabilité d'un clustering sur un ensemble d'observations consistant chacune en l'amplitude des partiels d'une note pendant une trame. Cette approche consiste donc dans un premier temps en l'élaboration et l'évaluation d'une mesure de similarité entre des

1. Nous prenons le parti comme bon nombre d'articles en français traitant de la « formation non supervisée de classes » d'utiliser le terme anglais correspondant : clustering »

trames de signaux harmoniques monophoniques, ne prenant pas en compte, ou de manière très simplifiée, l'évoluation temporelle globale.

Une seconde approche consiste à inclure la séparation ou clustering des notes présentes en fonction des sources dans l'algorithme multipitch lui même, i.e réaliser l'analyse multipitch et la séparation de source ensemble. Cette approche même si très élégante en théorie s'avère difficile à mettre en oeuvre, et nous montrerons qu'à l'heure actuelle une analyse multipitch suivie dans un second temps d'un clustering des notes ou même des trames donne de meilleurs résultats.

Dans une première partie de ce rapport non étudierons quelques algorithmes déjà existants, ensuite nous apporterons nos contributions basées sur une modèlisation source filtre des amplitudes de partiels, ou encore sur une mesure de similarité et les méthodes à noyaux.

2 Termes, Notations

- le terme X est une matrice d'observation signifie que X est une matrice $N \times D$ contenant sur chaque ligne un vecteur de features (ou ensemble de descripteurs) définissant un point dans $\mathbb{R}^D$. Souvent un de ces vecteurs sera une suite d'amplitudes de partiels ou une enveloppe fréquentielle estimée par interpolation et discrétisée en fréquence.
- $\phi(x)$ désigne l'image par une transformation non linéaire, engendrée par une fonction noyau, d'une observation.
- si $X : n \times m$ est une matrice réelle, $||X||$ représente la norme de Frobenius (la norme euclidienne au sens vectoriel) :

$$||X|| = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} X_{ij}^2}$$

- si $M : n \times m$ est une matrice, $X_{.n}$ et $X_{m.}$ représentent respectivement la $n$ème colonne et la $m$ème ligne de X.
- la lettre J est utilisée plus d'une fois dans ce rapport. Dans tous les cas il s'agit d'un critère à minimiser. Lorsque ce critère est quadratique la gradient du critère est calculé et annulé pour permettre d'exprimer les solutions.
- concernant les K-means dans l'espace des enveloppes, nous avons tenté d'uniformiser les notations :
  - x_i est un vecteur d'amplitudes de partiels, compressées ou non
  - e_c, D_c sont des attributs d'une certaine classe c
  - e_c est vecteur représentant l'enveloppe qui génére les observations de la classe c. L'élément $e_c(k)$ de ce vecteur représente l'amplitude de l'enveloppe à une certaine fréquence f_k. En général l'axe des fréquences a été échantillonné linéairement.
  - d_c est un vecteur représentant la pondération dépendante du numéro de partiel. $D_c = \text{diag}(d_c)$ est une matrice diagonale utilisée pour simplifier les écritures.
  - H est une matrice Toeplitz de filtrage passe bas. Dans nos programmes Matlab nous avons utilisé : H = toeplitz([blackman(21)', zeros(1,NbFreq - 21)]
  - $\Pi_i : NbFreq \times P$ est une matrice indicatrice des fréquences des partiels de la note i. $NbFreq$ est le nombre de fréquences où l'on échantillonne l'enveloppe, P est le nombre de partiel de la note i. Ainsi si les partiels tombent exactement sur des fréquences où l'on échantillonne on aura $\Pi_i(k, p) = 1$ si la fréquence du partiel p est f_k, 0 sinon. Sinon nous utilisons une interpolation linéaire ce qui ne nous semble pas être une approximation dérangeante si l'enveloppe est suffisament lisse.
  - $M_i : (NbFreq + P) \times NbFreq$ est la matrice comprenant dans sa partie supérieure $H\Pi_i$, plusieurs lignes de 0 si $P < length(d_c)$ et dans sa partie inférieur la matrice identité.
- parfois un vecteur sera un vecteur ligne, d'autre fois non. Généralement il est multiplié par une matrice ce qui permet de savoir dans quel cas on se trouve. Certaines formules matricielles sont applicables également aux vecteurs, si bien que volontairement on ne précise pas s'il s'agit d'un vecteur ou non.
- sur tous les graphiques montrant des résultats en clustering, les pourcentages affichés sont des pourcentages d'erreur.

3 Modèle sinusoïdal

Un son est un signal fonction du temps, échantilloné à plusieurs dizaines de milliers de Hertz. Néanmoins, pour les sons quasi-harmoniques, la redondance des informations est telle que la dimensionnalité peut-être fortement réduite après projection sur une base temps-fréquence telle que celle de la TFCT (transformée de Fourier à court terme), et

sélection des pics d'énergie avec suivi (ou « tracking ») des trajectoires dans le plan temps fréquence. On écrit alors que :

$$x(n) = \Re\left(\sum_{p=1}^{P} \tilde{A}_{part}(n,p) e^{2i\pi \tilde{F}_{part}(n,p)n}\right) + e(n)$$

où $e(n)$ est un signal résiduel supposé peu harmonique et d'énergie faible.

On impose de plus que :
- les fréquences des partiels soient approximativement multiples de la fréquence fondamentale, ou que l'écart entre les fréquences de 2 partiels consécutifs soit approximativement égal à la fréquence fondamentale :

$$\tilde{F}_{part}(n,p) - \tilde{F}_{part}(n,p-1) \approx 2\tilde{F}_{part}(n,1)$$

- la fréquence et l'amplitude évolue lentement dans le temps

$$\tilde{F}_{part}(n,p) \approx \tilde{F}_{part}(n-1,p) \qquad \tilde{A}_{part}(n,p) \approx \tilde{A}_{part}(n-1,p)$$

On peut ainsi sous-échantilloner en temps et écrire que

$$\tilde{A}_{part}(n,p) = [A_{part}(.,p)^{\uparrow} * \ h](n) \qquad \tilde{F}_{part}(n,p) = [F_{part}(.,p)^{\uparrow} * \ h](n)$$

Un son quasi-harmonique est alors défini par 2 signaux bidimensionnels :
$A_{part}(t,p)$ qui associe à chaque temps et chaque numéro de partiel une amplitude complexe, et $F_{part}(t,p)$ qui associe à chaque temps et chaque numéro de partiel une fréquence réelle positive.

On passe ainsi du plan temps-fréquence à une grille trame-harmonique (après séparatation/tracking des sources dans le cas de signaux polyphoniques).

Les paramètres du modèle sont estimés à l'aide d'un banc de filtre, par exemple la TFCT, et souvent seul le module des amplitudes est retenue, la phase ne contenant que peu d'informations pertinentes la fréquence précise étant déjà connue / estimée.

Le modèle sinusoïdal pour les sons polyphoniques consiste alors à supposer qu'un son quelconque $y(n)$ est la somme de plusieurs sons quasi-harmoniques (ou sources) $x_s(n)$ ajoutée d'un résiduel :

$$y(n) = \sum_{s=1}^{S} x_s(n) = \sum_{s=1}^{S} \Re\left(\sum_{p=1}^{P} \tilde{A}_{part}(n,p,s) e^{2i\pi \tilde{F}_{part}(n,p,s)n}\right) + e(n)$$

4 Modèle source-filtre, enveloppe, interpolation des partiels

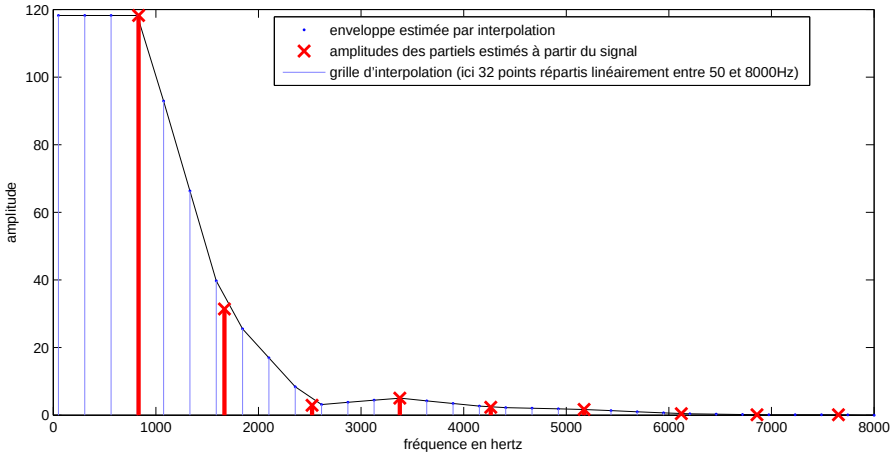

FIGURE 2 – Exemple d'enveloppe obtenue par interpolation linéaire des amplitudes de partiels. Trame de 100ms, 100ms après l'attaque d'un Ab5 ($f_0 = 830$Hz) de piano.

Dans le cas d'instruments réels, le modèle source-filtre permet d'exprimer simplement les spécifités du résonnateur qui ne dépendent pas de la fréquence de la note jouée. On suppose donc qu'il existe un filtre propre au résonnateur a priori de réponse en fréquence très régulière :

$$X(n,f) \simeq S(n,f)Env(n,f)$$

$X(n,f)$ est la TFCT d'un son harmonique, le spectre excitateur $S(n,f)$ est supposé ayant une forme de peigne, de fréquence fondamentale $f_0(n)$ localement constante en temps, et d'amplitude variable sur l'axe en temps mais constante sur l'axe des fréquences :

$$S(n,f) = A_n \sum_{p=1}^{P} \delta(f - pf_0(n))$$

A partir d'un son (par exemple de la base de donnée RWC [10, RWC2003] que nous avons beaucoup utilisée dans ce stage), on peut mesurer l'amplitude des partiels pour chaque trame, i.e réaliser une analyse sinusoïdale. L'enveloppe est ensuite estimée en interpolant ces amplitudes en fréquence. Ce qui nous amène à une des représentations utilisée durant ce stage : représentation sous forme d'enveloppe fréquentielle, estimée en interpolant l'amplitude compressée, ou non, des partiels (voir figure 2).

5 Modèle source-filtre et modèle acoustique

Le modèle source-filtre est un modèle simplifié où la source est supposée périodique, et les partiels de la source sont supposés être d'amplitude approximativement constante. L'amplitude du partiel p de la source s'écrit alors :

$$A_p = A$$

Pourtant, dans le cas réel, on observe souvent une décroissance avec la fréquence des amplitudes des partiels de la source. En acoustique, dans le cas d'une corde pincée avec dissipation on écrit généralement :

$$A_p \simeq \frac{A}{\sqrt{p}}$$

On écrit alors le spectre de la source :

$$S(n,f) = A_n \sum_{p=1}^{P} \frac{1}{\sqrt{(f/f_0)}} \delta(f - pf_0)$$

si bien qu'en généralisant à tous les modes de dissipation (à toutes les pondérations des partiels) on obtient :

$$S(n,f) = A_n \sum_{p=1}^{P} D_p \delta(f - pf_0)$$

Avec toujours le même modèle de signal :

$$X(n,f) = S(n,f)Env(n,f)$$

Ce modèle source-filtre modifié a été étudié durant la fin ce stage, et nous prévoyons encore plusieurs expériences pour valider ou invalider l'intéret de ce modèle dans le cadre de la formation de source. La section 19.4 décrit un algorithme ressemblant aux K-means intégrant ce modèle.

Deuxième partie
Modèles et etat de l'art

6 Travaux en reconnaissance d'instrument

Nous présentons succinctement deux travaux en reconnaissance d'instrument.

Disposant de deux bases de données, l'une utilisée pour l'aprrentissage et l'autre pour le test, comprenant des échantillons sonores (« samples ») de plus de quinze instruments, un ensemble de descripteurs de signaux sont calculés. Les descripteurs calculés à partir de la base d'apprentissage sont utilisés pour optimiser les paramètres d'un classifieur, celui-ci pouvant utiliser notamment :

– une taxonomie des instruments (classifieur hiérarchique)

– une réduction de dimensionnalité supervisée de type LDA
– une réduction de dimensionnalité non supervisée de type PCA
et être de type :
– KNN (« K-neareast neighbours »)
– classifieur gaussien [2]
– SVM (« Suppport Vector Machine »)

Les performances atteintes par les 2 systèmes présentés ci-dessous, sont comprises entre 70% et 80%. Les 2 bases de données étant différentes, ces résultats sont à prendre avec précaution, néanmoins ils montrent l'intérêt d'utiliser un grand nombre de descripteurs et d'une sélection manuelle ou automatique des descripteurs pertinents pour améliorer les performances en classification.

6.1 Travaux de Eronen et Al.

Voici les descripteurs utilisés par [3, Eronen2000] :

1	Rise time, i.e., the duration of attack
2	Slope of line fitted into rms-energy curve after attack
3	Mean square error of line fit in 2
4	Decay time
5	Time between the end of attack and the maximum of rms-energy
6	Crest factor, i.e., max / rms of amplitude
7	Maximum of normalized spectral centroid
8	Mean of normalized spectral centroid
9	Mean of spectral centroid
10	Standard deviation of spectral centroid
11	Standard deviation of normalized spectral centroid
12	Frequency of amplitude modulation, range 4-8Hz
13	Strength of amplitude modulation, range 4-8Hz
14	Heuristic strength of the amplitude modulation in range 4-8Hz
15	Frequency of amplitude modulation, range 10-40Hz
16	Strength of amplitude modulation, range 10-40Hz
17	Mean error of the fit between each of steady state, intensities and mean steady state intensity
18	Mean error of fit between each of onset intensities and mean onset intensity
19	Overall variation of intensities at each band
20	Fundamental frequency
21	Standard deviation of fundamental frequency
22-32	Average cepstral coefficients during onset
33-43	Average cepstral coefficients after onset

A partir d'une taxonomie des instruments de musiques présents dans la base de donnée, des classifieurs plus simples (le classifieur de niveau le plus élevé discrimine simplement les instruments à son entrenu [3] des autres) sont construits à partir de la base de test. Ceux-ci sont soit des classifieurs gaussiens soit des classifieurs de type KNN, suivant le nombre de classes à discriminer, le classifieur gaussien montrant de meilleurs performances pour la classification binaire.

Pour chaque classifieur, les descripteurs les plus discriminant sont sélectionnés manuellement ...

6.2 Travaux de Kitahara et Al.

Kitataha et Al. [14, Kitahara2003] procède d'une façon assez originale pour effectuer la reconnaissance d'instrument.

Il commence par calculer un ensemble assez important de « features » pour chaque trame d'une base d'apprentissage. Après avoir normalisé chaque descripteur, il opère ensuite une réduction de dimensionnalité non supervisée à l'aide de PCA , et enfin une réduction de dimensionnalité supervisée ou discriminante (voir le paragraphe « Linear Discriminant Analysis » 28).

Il obtient donc un ensemble très réduit de descripteurs qui sont chacun des combinaisons linéaires des descripteurs originaux. Pour réaliser la classification, il modélise une observation x de la classes c et de fréquence fondamentale f_0 comme : $x = \mu_c(f_0) + \Sigma^{1/2}\epsilon$

où ϵ est une variable normale multidimensionnelle centrée réduite ($\mathbb{E}(\epsilon\epsilon^T) = I$).

2. La plupart du temps il s'agit d'une décision Bayesienne appliquée à un modèle GMM

3. Les vents et le violon sont des instruments à son entrenu. Le piano, la guitare acoustique n'en sont pas. Mais cette classification pose des problèmes, dans un morceau de musique il arrive qu'un son de piano dure plus longtemps qu'une note de saxophone, ce qui amène à considérer des descripteurs de signaux qui vont par exemple réaliser une régression sur l'enveloppe temporelle du signal pour déterminer s'il y a une décroissance exponentielle, des mouvements dus à l'instrumentiste, etc.

Le « centroïde » μ d'une classe dépend de la fréquence fondamentale.

Pour estimer la fonction $\mu(f_0) \to \mathbb{R} \to \mathbb{R}^d$ où d est le nombre de descripteurs, Kitahara réalise une régression [4] (des moindre carrés) polynomiale de degré 3, et ainsi $\mu(f_0)$ est une fonction [5] très lisse en fréquence . Une fois les centroïdes connus, la matrice de covariance peut-être estimée, celle-ci étant supposée indépendante de la fréquence fondamentale.

Les descripteurs (ou « features ») utilisés par [14, Kitahara2003] sont les suivants :

(1) Spectral features (40 features) e.g., Spectral centroid, Relative power of the fundamental component, Relative power in odd and even components
(2) Temporal features (35 features) e.g., Gradient of a straight line approximating power envelope, Average differential of power envelope during onset
(3) Modulation features (32 features) e.g., Amplitude and frequency of AM, FM, modulation of spectral centroid and modulation of MFCC
(4) Non-harmonic component features (22 features) e.g., Temporal mean of kurtosis of spectral peaks of each harmonic component (Their values become lower as sounds contain more non-harmonic components.)

6.3 Conclusion sur la classification d'instrument

Les performances atteintes par ces systèmes de classification sont impressionnantes, mais les descripteurs utilisés sont nombreux, peu robuste en cas de mélange polyphonique. Des méthodes de sélections de features sont indispensables, et les performances dépendent en très grande partie de comment ces features ont été sélectionnées / pondérées.

Dans ce stage nous avons pris le parti de travailler avec un nombre très réduit de descripteurs, à savoir uniquement les amplitudes des partiels, car ceux-ci sont faciles à visualiser, beaucoup de travail peut déjà être réalisé pour construire des mesures de similarités plus performantes que la simple distance euclidienne entre les vecteurs d'amplitudes ou les MFCC. Atteindre des performances équivalentes en classification, en reprenant ces articles fait partie des projets possibles dans la suite de ce stage.

7 Travaux de Heittola et Al.

Heittola et Al. [1, Heittola2009] propose une méthode originale pour la formation de source en polyphonique, après analyse multipitch. Pour cela il utilise un modèle équivalent à ce que nous décrivons dans le paragraphe sur K-means dans l'espace des enveloppes 19.3, et utilise un algorithme équivalent à ce que nous appelons NMF déformé 26.

Il utilise un modèle source-filtre où le nombre de filtres correspond au nombres d'instruments I, et le nombre de sources N à la polyphonie existante à chaque trame. Ainsi l'amplitude du spectre observé à une trame est approximé par :

$$\tilde{X}(k,t) = \sum_{n=1}^{N} \sum_{i=1}^{I} g_{n,i}(t) S_i(k) e_n(k,t) \quad (eq.58)$$

où k et t sont les indices fréquentiels et temporels $S_i(k) = \sum_{m=1}^{M} a_m(k) c_{m,i}$ est le modèle d'enveloppe de l'instrument i, modélisé comme la somme pondérée de filtres passe-bande triangulaires $a_m(k)$.

$e_n(k,t)$ est le spectre excitateur de la note n à la trame t correspondant à un peigne harmonique défini par une amplitude et une fréquence fondamentale, et optionnellement un facteur d'inharmonicité (ce modèle est donc plutôt inspiré du modèle source-filtre).

---

4. Il est possible de modéliser $\mu(f_0)$ d'une manière plus flexible, inspirée des calculs réalisés au paragraphe traitant de l'algorithme K-means dans l'espace dans enveloppes '19.3.

Soit un ensemble d'observations $x_1 \cdots x_N$ issues d'une même classe et de fréquences fondamentales $f_0^1 \cdots f_0^N$.

Soit E la matrice contenant sur une ligne le centroïde d'une certaine fréquence fondamentale.

Il existe des vecteurs lignes $\pi_1 \cdots \pi_N$ indicateurs des fréquences fondamentales tel que :

$x_i \simeq \pi_i E$

Pour assurer que les centroïdes sont lisses en fréquence (dans le sens des colonnes de E), on remplace E par HE où H est une matrice de stucture Toeplitz de filtrage passe bas.

Enfin on ajoute un terme de régularisation, et on trouve le critère suivant à minimiser par rapport à E : $J = \sum_i ||x_i - \pi_i HE||^2 + \gamma ||E||^2$ Ce qui donne : $E_{.p} = (H^T \sum_{i,p} \pi_i^T \pi_i H + \gamma I)^{-1} (\sum_{i,p} x_{ip} H \pi_i^T)$

5. La figure donnée dans l'article montre sur un exemple une répartition très régulière des descripteurs autour de cette fonction et est donc très impressionnante. Il ne s'agit que d'un exemple mais il serait intéressant de « creuser » car une telle répartition pourrait permettre d'espérer obtenir de bons résultats en clustering.

$g_{n,i}(t)$ est le gain de l'instrument i pour la note n de la trame t.

Après analyse multipitch, les paramètres à estimer sont donc $g_{n,i}(t)$ et $c_{m,i}$.

Ceci est résolu par un algorithme proche de NMF [6] , ces paramètres étant supposés non négatifs.

Le nombre I de modèles de filtres ou d'instruments est limité pour tout le morceau, si bien que cet algorithme réalise un clustering flou ($g_{n,i}$ doit être compris comme l'affinité de la note n pour le modèle d'enveloppe i) variant au cours d'une note ($g_{n,i}$ dépend de t). Pour effectuer un clustering complet, Klapuri et Al. propose de classer chaque

note en fonction du maximum de vraissemblance vis à vis de modèles GMM [7] élaborés pour chaque instrument réels. Les features utilisées pour décrire les trames sont les coefficients (δ-)MFCC. Les évaluations montrent une F-measure [8]

de l'ordre de 55%.

8 Travaux de Miyamoto et Al.

Miyamoto et Al. [9, Miyamoto2008] utilise une approche différente où le modèle de timbre correspond à l'amplitude relative des partiels. Un aspect relativement nouveau est que l'analyse multipitch et le clustering (possiblement flou) des notes sont réalisés en même temps, mettant ainsi en évidence l'aspect de complémentarité de ces deux tâches.

La distribution énergétique temps fréquence observée sur le signal est supposée être la somme de k sources harmoniques de fréquence fondamentale f_k, d'amplitude a_k constante, et appartenant chacune à une classe de timbre c_k.

Les notes d'une classe de timbre possèdent des propriétés communes : l'énergie relative des différents partiels est la même et ne varie pas dans le temps, et l'évolution de l'amplitude temporelle (donc ne variant pas en fréquence) est la même pour toutes ces notes. Le modèle de timbre sous-jacent (ici plutôt inspiré du modèle sinusoïdal) est donc de dimension $T + P$ où T est le nombre de points définissant l'évolution de l'amplitude temporelle, et P le nombre de partiels. Ce modèle naïf est ensuite transformé en un modèle probabiliste, et le template de base rigide est relaxé en une distribution GMM dans le plan temps-fréquence. Un algorithme itératif de type «Expectation Minimization » (EM) est enfin utilisé pour estimer les paramètres du modèle.

Cette approche présente l'avantage de régler avec un unique algorithme l'analyse multipitch, l'estimation des modèles des sources harmoniques et le clustering, le problème des partiels superposés ... Elle présente cependant l'inconvéniant d'être difficilement améliorable, et soufre de l'existence d'un grand nombre de minimum locaux lors de la maximisation du critère décrit dans l'article. Enfin on peut se demander si le modèle de timbre utilisé n'est pas trop simplifié pour traiter correctement certain cas problématiques.

Les résultats ne sont pas chiffrés et seul un résultat sur un extrait piano-violon est donné, celui-ci étant plutôt bon pour l'analyse multipitch, mais moins convaincant pour le clustering des notes.

9 Travaux de Burred, modèle temporel simple

Le modèle utilisé est le suivant : pour chaque instrument, une note est supposée être issue d'une distribution gaussienne dont les paramètres μ et Σ dépendent du temps. Il s'agit d'un article qui traite de classification, mais nous avons préféré le placer ici car contrairement à Kitahara et Al. et Eronen et Al. il n'utilise pas d'autres descripteurs que les amplitudes de partiels.

Sur une base de donnée de sons d'un même instrument, après analyse sinusoïdale, et interpolation des amplitudes de partiels mesurés sur chaque trame , on obtient un ensemble de trajectoires chacune étant une suite temporelle d'enveloppes.

Ces trajectoires peuvent être alignées temporellement selon différent critères, le plus simple étant de compter sur le fait que les durées des sons de la base de donnée ont déjà été rendues cohérentes. De la même manière, on suppose dans un premier temps que la normalisation en intensité a été réalisée correctement.

Pour chaque indice de temps r les enveloppes sont supposées suivre une distribution multivariée gaussienne la probabilité d'une observation s'écrit donc : (après avoir centré les observations $x \leftarrow x - \mu$)

$$p(x) = e^{-x\Sigma^{-1}x^T}$$

6. Non Negative Matrix Factorization. Il s'agit d'un alogrithme de factorisation de matrice non négative en matrices de plus petites dimensions également à coefficients positifs. Les algorithmes existant sont tous du type : algorithme itératif de minimisation alternée avec contraintes.

7. Gaussian Mixture Model, densité de probabilité possiblement multidimensionnelle qui s'écrit comme une somme de gaussiennes dans le cas général multivariées.

8. indice de performance calculé à partir du nombre des proportions de faux négatifs et de faux positifs

où Σ est la matrice d'autocovariance. On suppose que celle-ci est inversible, si ce n'est pas le cas, il est courant d'utiliser un seuil : $\Sigma \leftarrow \Sigma + \epsilon Id$

Ne disposant pas de la matrice d'autocovariance réelle, on doit l'estimer à partir des observations. On utilise pour cela un estimateur non biaisé :

$$\widehat{\Sigma} = X^T X/(N-1)$$

on peut au préalable projeter les observations dans l'espace PCA estimé à partir des observations, où la matrice d'autocovariance, projetée elle-aussi, devient approximativement diagonale (voir paragraphe 29)

$$p(x) = e^{-(xP)\Lambda^{-1}(xP)^T}$$

$$\Lambda = P^T X^T X P/(N-1) = P^T \widehat{\Sigma} P$$

est la matrice d'autocorrélation estimée dans l'espace PCA

Le modèle proposé par Juan Jose Burred [2, Burred2009], qui intègre l'évolution temporelle, est le suivant :

les trajectoires de K instruments, issues de plusieurs notes d'une même octave et de plusieurs modes de jeu / instrumentistes sont centrées puis projetées dans l'espace PCA, calculé à partir de l'ensemble des trames observées.

ensuite, pour chaque instrument et pour chaque index temporel, la variance dans chacune des composantes est calculée : $\tilde{\Lambda} = \text{diag}(\Lambda) = \text{diag}(P^T X^T X P/(N-1))$

enfin, les variances sont projetées dans l'espace original, on suppose à nouveau que les composantes sont décorrélées, et on obtient donc :

$$p(x|\mu_{k,r}, \tilde{\Sigma}_{k,r}) = \exp(-(x-\mu_{k,r})\tilde{\Sigma}_{k,r}(x-\mu_{k,r})^T)$$

avec μ_k la moyenne des observations de l'instrument k à l'index temporel r et

$$\tilde{\Sigma}_{k,r} = \text{diag}\left(P^T \,\text{diag}(P^T X_{k,r}^T X_{k,r} P/(N-1))P\right)$$

la vraissemblance d'une classe au regard d'une trajectoire s'écrit alors :

$$\mathcal{L}(k\ ; x, \mu_{k,r}, \tilde{\Sigma}_{k,r}) = \prod_{t=1}^{T} p(x|\mu_{k,r}, \tilde{\Sigma}_{k,r})$$

on peut ajouter, au préalable, une normalisation en intensité, une compression des amplitudes, une normalisation en durée, ou encore une segmentation de type « Attack Decay Sustain Release » (ADSR) pour affiner le modèle (voir [5, Burred2009b]).

9.1 Essais expérimentaux

On modélise donc, pour chaque instrument, l'enveloppe fréquentielle observée à l'index temporel t par une distribution gaussienne de matrice d'autocovariance diagonale dans l'espace PCA. On suppose qu'il existe une distribution gaussienne pour chaque couple instrument/index temporel.

Pour assurer que la matrice d'autocovariance soit diagonale, il parait aproprié de calculer la transformation PCA pour chacun de ces couples.

Le problème est que si les bases PCA sont différentes d'un index temporel à l'autre et d'un instrument à l'autre, ce qui n'a que peu de raisons de ne pas être le cas, alors il ne sera plus possible de visualiser facilement (dans l'espace des 3 premières composantes principales) les trajectoires et les variances.

Il est donc plus commode de projeter toutes les trames de tous les instrument dans un même espace de dimension réduite, et d'estimer la matrice d'autocovariance dans cet espace même si celle-ci n'est pas diagonale.

pseudo-boule : volume défini par un centre c et une matrice M, dont le contour, translaté à l'origine, correspond à l'image par une application linéaire d'une hypersphère de rayon 1. Le terme hyper-ellipse est également parfois employé car les contours des projections d'un tel volume sur des plans sont des ellipses.

$$V = \{x \mid \exists y, ||y|| \leq 1, x = c + yM\}$$

On peut également définir, si M est symétrique, avec la distance de mahanalobis $d_M(x,y)^2 = xM^2y^T$, une pseudo-boule comme étant une boule généralisée de rayon 1 et de centre c pour la distance d_M.

$$V = \{x \mid d(x,c) \leq 1\}$$

Si on construit les pseudo-boules engendrées par la racine carré de la matrice d'autocovariance des observations, on peut « voir » la variance des « prototypes curves » dans les trois premières composantes principales. Ces « prototypes

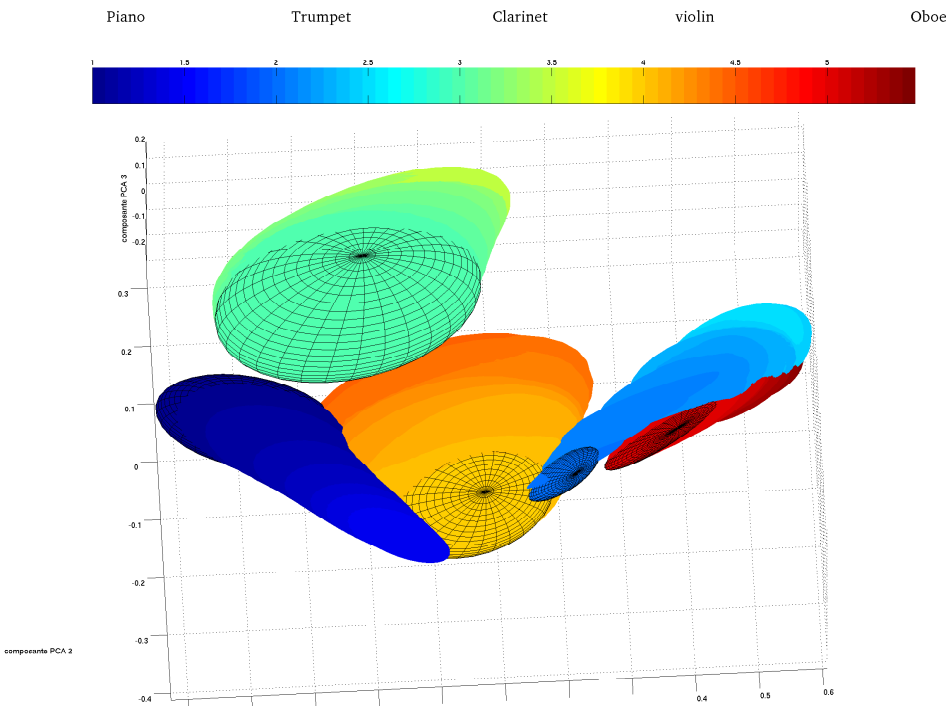

FIGURE 3 – pseudo-boules d'autocovariance dans les 3 premières composantes PCA

curves » sont les trajectoires temporelles moyennes des notes d'un même instrument. La variance que l'on représente est donc celle existente d'une note à l'autre, et pour chaque instrument il y a une quinzaine de boules, une pour chaque index temporel.

Comme pour les distributions mono-dimensionnelles, ces pseudo-boules contiennent en moyenne 68% des observations d'un certain instrument à un certain index de temps, notamment si les observations sont supposées suivre une distribution gaussienne.

Les boules quadrillées représentent l'attaque, et ont donc été construites sur la première trame des trajectoires extraites des samples RWC. On remarque que cette partie d'attaque seule est fortement discriminante, permettant probablement de dépasser les 60% de classification correcte dans un cadre de reconnaissance d'instrument à partir des trajectoires complètes. L'ennui est que ces parties d'attaque sont justement celles qu'on ne sait pas extraire à partir de mélange polyphonique.

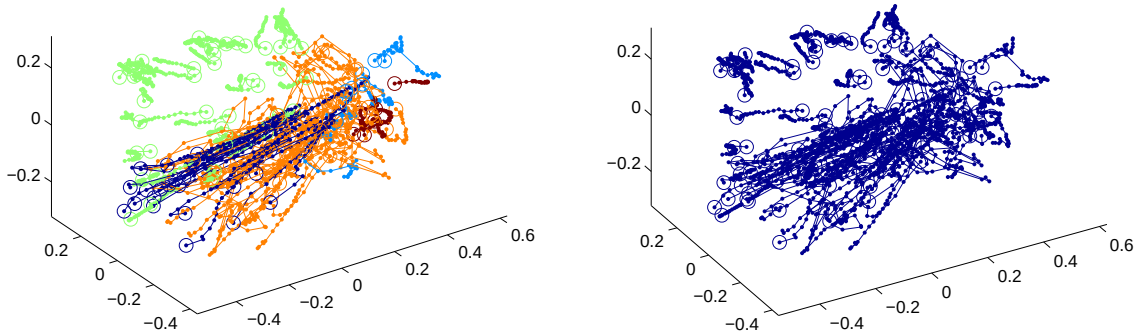

FIGURE 4 – trajectoires des notes dans l'espace PCA, étiquettées puis non étiquettées

Sur la figure reftrajectoiresPCA, on peut voir à gauche les trajectoires dans l'espace PCA, et à droite les même trajectoires telles qu'un algorithme de clustering les connait c'est à dire non étiquettées. On voit clairement que la

tâche est difficile en utilisant cette représentation.

10 Travaux de Graham Grindlay et Dan Ellis

Dans [6, Grindlay2009], une méthode originale mélangeant plusieurs des approches décrites plus haut est présentée, même si celle-ci reprend l'idée de Miyamoto de réaliser avec un seul et même algorithme la transcription, l'élaboration des modèles de sources, et l'affectation de chaque note à une source.

Pour chaque « pitch » [9] , les spectres d'amplitudes d'un certain ensemble d'instruments (30 dans l'article) sont calculés, et une décomposition NMF est réalisée. Les vecteurs de base ainsi obtenus sont utilisés pour décomposer les spectres d'amplitudes de sons polyphoniques où les modèles d'instruments sont inconnus, mais dont le nombre est fixé.

Une note d'un nouvel instrument sera modélisée comme étant la somme pondérée des vecteurs de base obtenus précédemment. Cette décomposition du timbre d'un nouvel instrument sur des vecteurs de base obtenus par apprentissage est dénommée « eigeninstrument decomposition », en référence à la « eigenvoice decomposition » très utilisée en traitement de la parole.

Pour que les coefficients de pondération des différents pitch d'une même source soient cohérents, la première étape est un peu modifée :

pour chaque instrument i, pour chaque pitch p, N spectres d'amplitude de longueur F sont calculés, et empilés sous la forme d'une matrice $M = IN \times PF$

ensuite l'algorithme NMF est utilisé pour trouver R vecteurs de bases $W_1 \cdots W_R$ tels qu'une ligne $M_{i,n}$ de M (qui contient donc P spectres d'amplitude mis bout à bout) peut être approximée par :

$$M_{i,n} \approx \sum_{r=1}^{R} h_{i,n,r} W_r$$

un spectre d'amplitude $U_{i,n,p}$ composant la ligne $M_{i,n}$ peut donc être approximé par

$$U_{i,n,p} \approx \sum_{r=1}^{R} h_{i,n,r} w_{r,p}$$

avec $W_r = [w_{r,1} \cdots w_{r,P}]$. Enfin, le spectre d'amplitude V_n d'une trame d'un morceau polyphonique peut être approximé par

$$V_n \approx \sum_{s=1}^{S} \sum_{r=1}^{R} g_{s,r,n} \alpha_{s,p,n} w_{r,p}$$

En utilisant un algorithme proche de NMF, il est possible d'optimiser les paramètres g et α, dans l'article le critère uilisé est la divergence Itakura-Saito, ou plutôt une relaxation de celle-ci. Le paramètre $\alpha_{s,p,n}$ doit être interprété comme « l'instensité du pitch p joué par l'instrument s à la trame n ».

11 Conclusion

Dans les travaux de Grindlay, le modèle de timbre est le suivant : chaque source est modélisée comme une mixture strictement additive de vecteurs de bases, dont les coefficients de mélange sont les mêmes pour chaque pitch, mais les vecteurs de bases eux sont différents.

Dans les travaux de Myamoto, le modèle de timbre diffère uniquement en ce sens que les vecteurs de base correspondent à des pics seuls, et non pas des spectres d'amplitude. Chez Heittola, le modèle utilisé est explicitement un modèle source-filtre, où le filtre est une somme de formes de bases assez lisses.

Le défaut de la méthode de Myamoto est donc d'imposer le même modèle pour chaque pitch, et pour la méthode de Grindlay le défaut serait plutôt d'imposer que pour chaque pitch les coefficients de mélange soient exactement les mêmes, en plus du risque de ne pas extraire suffisament de vecteurs de bases où d'utiliser une base d'apprentissage qui ne soit pas assez générale.

Enfin, le modèle proposé par J.J. Burred suppose une distribution gaussienne des données, ce qui peut être avantageux et donner de bons résultats en classification, mais a peu de chance de permettre de bons résultats en clustering.

9. Pitch est un terme anglais qui peut se traduire par hauteur ou hauteur de note. Un « pitch » désigne ici une hauteur en échelle logarithmique, par exemple A#4 ou C4 .

Troisième partie

Elaboration d'une mesure de similarité de timbre pour les sons quasi-harmoniques

12 Présentation de la base de donnée utilisée

Nous utilisons les échantillons sonores (ou « samples ») fournis dans la base de donnée RWC [10, RWC2003]. Pour nos essais nous avons commencé par utiliser 13 instruments, puis nous avons décidé de réduire la base de donnée

à 6 instruments, les résultats présentés sont donc moins généralisables les résultats étaient beaucoup plus faciles à interpréter avec un nombre réduit d'instrument.

piano
violon
slap bass
trompette
clarinette en Bb
saxophone alto

Pour chacun de ces instruments nous disposons de 3 modes de jeux (piano, mezzo forte, forte) joués par 3 instrumentistes sur toute la tessiture de l'instrument.

Dans ce rapport voici ce que nous désignons comme « la base de donnée restreinte à l'octave 4 » : beaucoup de résultats donnés dans ce rapport ont été calculé en moyennant plusieurs essais réalisés sur une base de donnée « virtuelle » définie comme étant l'ensemble des combinaisons de 2 à 6 parmis ces 6 instruments, où pour chaque instrument un nombre aléatoire d'instrumentiste et de mode de jeu est sélectionné, puis un ensemble aléatoire d'au minimum 3 notes est sélectionné parmis les 12 de l'octave, et pour chaque note un nombre aléatoire de trame est sélectionné toujours 100ms après l'attaque.

D'autres résultats ont fait usage d'une base de donnée similaire mais comrenant les octaves 3,4,et 5.

Il serait intéressant de chiffrer les résultats dans le cadre de l'analyse multipitch, et c'était le principal objet de ce stage, mais nous n'avons pas élaboré de chaine de traitement complète (« framework d'évaluation ») comprenant :

- la synthèse à partir du Midi et d'une banque de sample,
- l'estimation multi-F0 (à réaliser pour chaque instrumentation de chaque morceau synthétisé),
- l'estimation des amplitudes partiels de chaque note à partir du mélange polyphonique et donc la résolution des partiels superposés
- et enfin l'étape de clustering / formation de sources / évualation.

Comme cela est souvent répété dans les articles traitant du sujet, beaucoup d'éléments peuvent introduire des erreurs, et il n'est donc pas inintéressant d'utiliser dans un premier temps des données idéales pour travailler uniquement sur la dernière étape. C'est ce que nous avons fait durant ce stage.

13 Observation empirique des données brutes

On remarque la figure 5 que l'amplitude d'un partiel de piano décroît généralement avec une pente linéaire en échelle logarithmique, avec un partiel 1 assez fort. Pour la clarinette on retrouve qu'à cette tessiture (reb4 à solb4) les partiels 1 et 3 sont fortement dominant par rapport au partiel 2. Les partiels de saxophone ont l'air également de suivre des règles simples, par contre pour la trompette et le violon, les courbes sont totalement chaotiques et on a peine à distinguer des caractères communs aux différentes notes, pourtant jouées par le même instrumentiste sur le même instrument. Pour la trompette, le changement d'harmonique de tube n'explique pas la différence importante entre les notes reb4 à mib4 et les autres, car il s'agit d'une trompette en sib qui change d'harmonique entre sol4 et lab4, donc entre fa4 et solb4 une fois transposé, alors que sur ce graphique ces 2 sont justement les plus similaires.

Cette figure a été placée en introduction de la partie sur les mesures de similarité de timbre pour avertir le lecteur sur la difficulté voir l'impossibilité d'établir une mesure de timbre idéale uniquement sur les données de partiels, et préfigure donc l'imperfection des mesures de similarités proposées dans ce rapport.

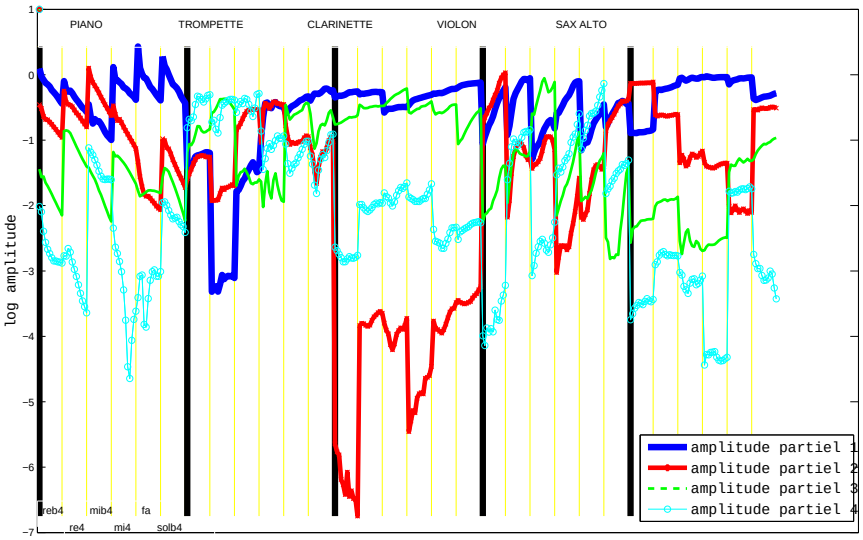

FIGURE 5 – évolution de l'amplitude des 4 premiers partiels des samples reb4 à solb4 pour 4 instruments, à chaque fois les 12 premières trames du sample sont gardées

14 Mesure de similarité de timbre entre deux sons quasi-stationnaires

Une mesure de similarité S associe une valeur comprise entre 0 et 1 à des couples d'objets appartenant à un ensemble $\mathcal{A}$. Quand $\mathcal{A}$ représente l'ensemble des enveloppes, un objet $a \in \mathcal{A}$ sera une fonction définie sur $[0; f_{max}]$ à valeur dans $[0; A_{max}]$.

Une mesure de similarité possible, que nous étudions dans ce rapport, sera la suivante :

$$S_{P,C}(a,b) = \exp(-||a-b||^2_{P,C}) \tag{1}$$

On reconnait un noyau gaussien et la norme L^2, paramétrée par une pondération P et une fonction de compression C :

$$||a-b||^2_{P,C} = \int_0^{f_{max}} (C(a(f)) - C(b(f)))^2 P(f) df \tag{2}$$

Nous montrerons que l'utilisation d'une fonction de compression est justifiée par l'amélioration qu'elle engendre en terme de clustering ou de classification. Cette fonction de compression pourrait être n'importe qu'elle fonction strictement croissante, dans ce stage nous nous limitons à :

$$C_{lin}(x) = x \quad C_{log}(x) = \log(\epsilon + x) \quad C_{exp}(x) = x^\gamma \tag{3}$$

Bien souvent les données sont normalisées avant d'être compressées, si bien qu'on ne s'intéresse à la forme de ces fonctions que dans l'intervalle $[0; 1]$.

Notons que les MFCC (Mel Frequency Cepstrum Coefficients) couramment utilisés en reconnaissance vocale, et parfois en reconnaissance d'instrument, font usage d'une compression logarithmique.

La pondération, présente avec le terme $P(f)$, permet quant à elle de donner un poids différent à certaines bandes de fréquences suivant n'importe qu'elle courbe de « Loudness », souvent définie comme la courbe moyennée sur plusieurs individus de sensation de puissance sonore équivalente à une excitation à 1000Hz de 40 dB.

14.1 Discrétisation de l'échelle des fréquences

Le théorème d'échantillonnage de shannon nous assure que si les enveloppes et la pondération P sont suffisament lisses (basses fréquences), la norme euclidienne définie entre les enveloppes échantillonnées restera inchangée (donc égale

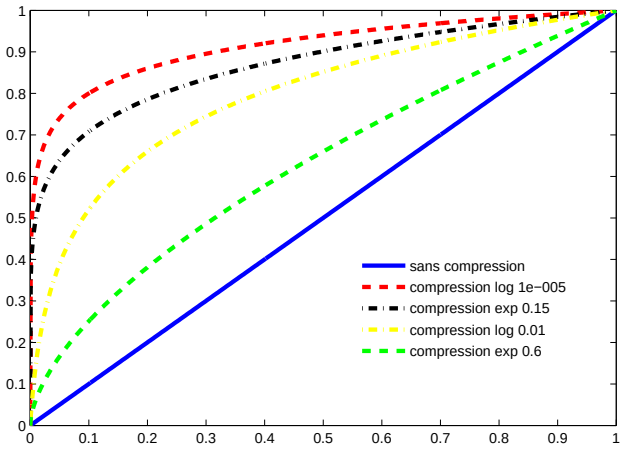

FIGURE 6 – quelques-unes des fonctions de compression utilisées dans ce rapport

à la norme L^2 entre les enveloppes continues). Soit s_i et s_j les enveloppes échantillonnées aux fréquences $f_0 \cdots f_K$.

On aura le théorème suivant : S'il existe une fonction g strictement croissante $g : [0; f_{max}] \to [0; f_{max}]$ avec $g(f_k) = kf_{max}/K$, tel que les fonctions $e_i(g^{-1}(u))\sqrt{P(g^{-1}(u))}$ eft $e_i(g^{-1}(u))\sqrt{P(g^{-1}(u))}$ respectent le théorème de shannon pour l'échantillonnage à f_{max}/K, alors :

$$\begin{aligned}
||s_i - s_j||^2 &= \sum_{k=1}^{K}(s_i(k) - s_j(k))^2 P(f_k) = \sum_{k=1}^{K}(e_i(f_k) - e_j(f_k))^2 P(f_k) \\
&= \sum_{k=1}^{K}(e_i(g^{-1}(k/K)) - e_j(g^{-1}(k/K)))^2 P(g^{-1}(k/K)) \\
&= \frac{K}{f_{max}} \int_0^{f_{max}} (e_i(g^{-1}(u)) - e_j(g^{-1}(u)))^2 P(g^{-1}(u)) du \\
&= \frac{K}{f_{max}} \int_0^{f_{max}} (e_i(f) - e_j(f)^2 P(f) g'(f) df \\
&= \frac{K}{f_{max}} \int_0^{f_{max}} (e_i(f) - e_j(f)^2 Q(f) df
\end{aligned}$$

avec le changement de variable $u = g(f)$ et en posant $Q(f) = P(f)g'(f)\frac{K}{f_{max}}$ On peut donc, en faisant la supposition que les enveloppes sont suffisament lisses et en choisissant correctement la pondération P et les fréquences où l'on échantillone, travailler sur les enveloppes échantillonnées au lieu des enveloppes réelles.

14.2 Autres mesures de similarité entre enveloppes

14.2.1 MFCC

Etapes pour calculer les MFCC :

1. normaliser l'enveloppe interpolée
2. calculer sur une echelle Mel la racine carré de l'énergie contenue dans chaque bande (filtres de réponse en forme de triangles de somme 1, de fréquence centrale calculée sur une échelle logarithmique et de largeur de bande proportionelle à la fréquence)
3. appliquer une compression *log*
4. calculer les 13 premier coefficients de la DCT

14.2.2 MFC-log, MFC-lin et MFC-exp

Nous proposons une alternative aux MFCC, les MFC-log, en changeant l'ordre des étapes.

Etapes pour calculer les MFC-log :

1. normaliser l'enveloppe interpolée
2. appliquer une compression *log*
3. calculer sur une echelle Mel la racine carré de l'énergie contenue dans chaque bande (filtres de réponse en forme de triangles de somme 1, de fréquence centrale calculée sur une échelle logarithmique et de largeur de bande proportionelle à la fréquence)
4. calculer les 13 premier coefficients de la DCT

en remplaçant la compression log par une compression exp ou en la retirant simplement on obtient les MFC-exp et les MFC-lin.

14.2.3 Entropie relative, divergence Beta, Itakura Saito, et Kullback Leibler

$$D_\beta(x||y) = \sum_i x_i \left(x_i \frac{x_i^{\beta-1} - y_i^{\beta-1}}{\beta(1-\beta)} + y_i^{\beta-1} \frac{y_i - x_i}{\beta} \right)$$

lorsque β tend vers 0 on trouve la divergence Itakura-Saito :

$$D_{ITS}(x||y) = \sum_i \frac{x_i}{y_i} - \log \frac{x_i}{y_i} - 1$$

lorsque β tend vers 1 on trouve la divergence Kullback-Leibler ou entropie relative :

$$D_{KL}(x||y) = \sum_i x_i \log \frac{x_i}{y_i}$$

Enfin lorsque $\beta = 2$ on trouve la distance Euclidienne.

En faisant varier β on peut donc faire un compromis entre la distance euclidienne qui est linéaire par facteur d'échelle et invariante par translation, et la divergence ITS qui est invariante par facteur d'échelle mais qui diminue par éloignement du zéros.

15 Présentation de quelques algorithmes utilisés

15.1 Principal Component Analysis

On peut constater que la variance des données dans la première composante principale est beaucoup plus grande que celles dans les 2ème et 3ème composantes.

15.2 Méthodes à Noyau

Le terme « méthode à Noyau (Kernel machine) » désigne une méthode pour rendre applicable les algorithmes de discrimination, classification ou transformation linéaire à des problèmes intrasèquement non linéaires. Parmis les principaux algorithmes linéaires souvent « Kernelisés » on peut citer :

- Support Vector Machine (SVM)
- Principal Component Ananylisis (PCA)
- Linear Discriminant Ananylisis (LDA)
- K-means

Il existe deux manières d'introduire les méthodes à noyau. La première plus théorique est développée dans l'annexe, nous présentons ici la seconde qui est tout à fait valable dans un cadre de clustering. Certaines fonctions de $\mathbb{R}^n \times \mathbb{R}^n$

dans $\mathbb{R}$, dites « fonction noyaux » ont la propriété de toujours donner une matrice semi définie positive (matrice diagonalisable dans une base orthonormée avec des valeurs propres postivies ou nulles) lorsqu'elles sont appliquées aux couples d'un ensemble de données. On ne discutera ici que du noyau gaussien. A partir d'un ensemble de N points de

$x_1 \cdots x_N$ de $\mathbb{R}^n$, on peut appliquer un noyau gaussien et calculer une matrice dite de similarité entre les points :

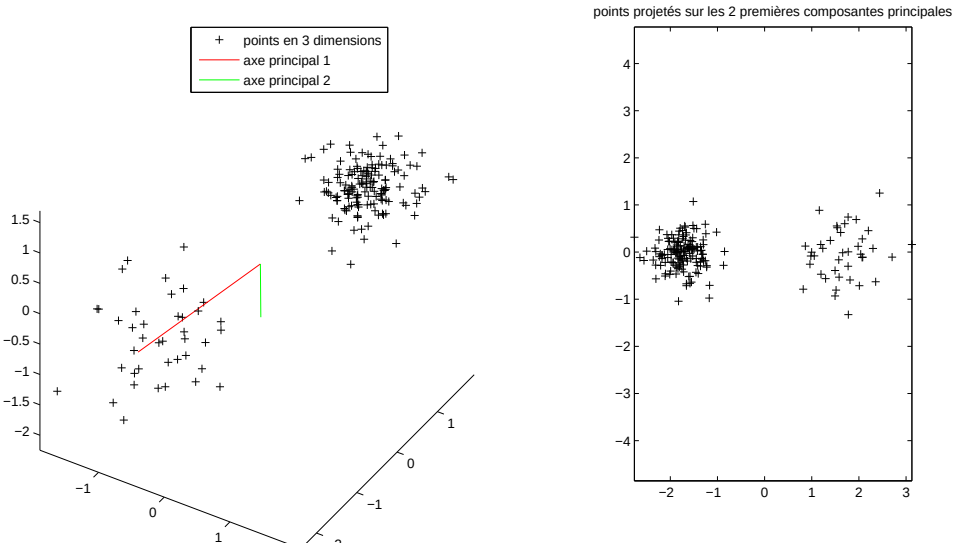

FIGURE 7 – Projection PCA

$$K_{ij} = \exp(-\lambda^2 ||x_i - x_j||^2)$$

Comme dit précédemment, le noyau gaussien a cette propriété que la matrice K est toujours semi-définie positive, quelque soit les points sur lesquels elle est calculée. On suppose ensuite que cette matrice est une matrice de Gram

(matrice des produits scalaires) dans un espace de dimension différente. C'est à dire que l'on suppose qu'il existe des points $y_i = \phi(x_i)$) et donc une transformation ϕ tel que $K(i,j) = \phi(x_i)\phi(x_j)^T$.

En diagonalisant la matrice K, on peut effectivement expliciter les $\phi(x_i)$. En écrivant $K = P\Lambda P^T$ et $Y = P\Lambda^{1/2}$ on a bien :

$$K = YY^T$$

La matrice Y, de dimension $N \times N$ contient alors dans ses lignes les coordonnées des y_i, et définit ainsi la transformation ϕ. Les y_i dans cet espace sont de dimension N, mais cela ne nous empêche pas de les « observer

», ou plutôt leur projection en 2 ou 3 dimensions, après avoir réalisé une transformation PCA. Cette transformation des données (application d'une fonction kernel, puis réduction de la dimensionnalité à l'aide de PCA) est désignée par le terme « Kernel PCA ».

15.3 K-means

voir en Annexe

15.4 Kernel K-means

Kernel K-means désigne un algorithme K-means transformé pour travailler dans l'espace kernel, donc il s'agit soit d'appliquer l'algorithme K-means habituel après diagonalisation de la matrice de similarité, soit d'utiliser le « kernel trick » qui évite l'étape coûteuse de diagonalisation de matrice.

voir en Annexe pour plus de détails

15.5 Normalized Cut

Normalized Cut est un critère de partionnement de graphe, qui peut également être appliqué à une matrice de similarité, et est dans ce cas un algorithme de clustering. En effet, le critère Normalized Cut est équivalent au critère « weighted kernel K-means » appliqué à une version normalisée de la matrice de similarité (appelée Laplacien dans la littérature) :

$$\mathcal{L} = I - D^{-1/2} K D^{-1/2}$$

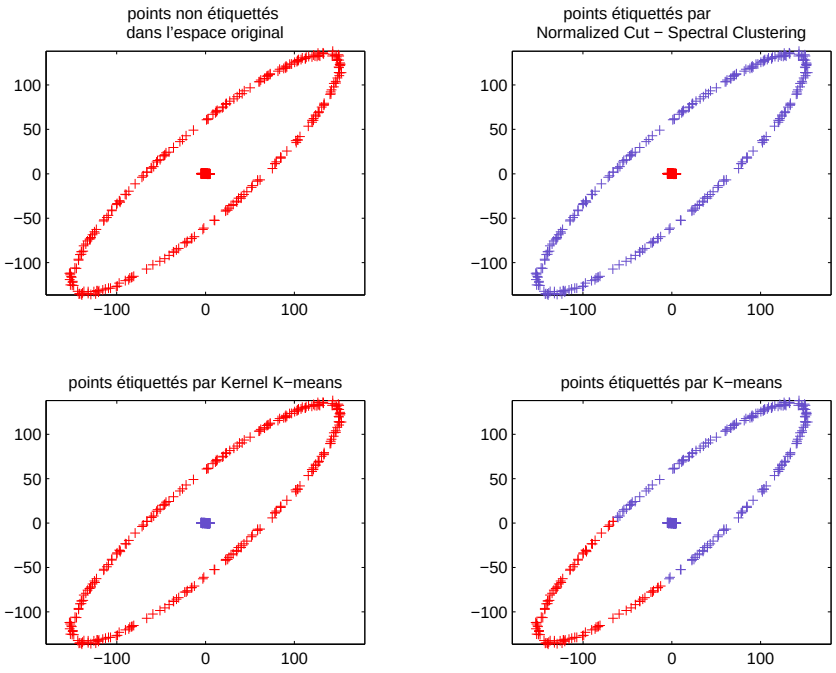

FIGURE 8 – Résultats de clustering pour 3 algorithmes

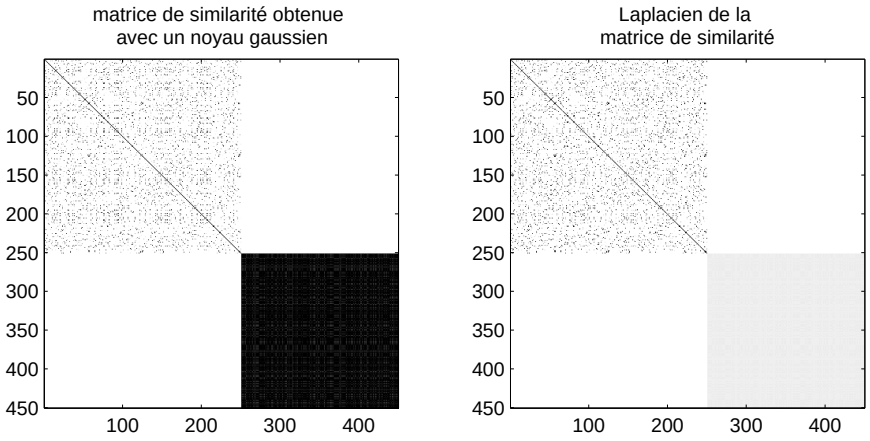

FIGURE 9 – matrices de similarités et matrice de similarité normalisée calulées à l'aide d'un noyau gaussien

que l'on remplace souvent par :

$$L = D^{-1/2} K D^{-1/2}$$

où K est la matrice de similarité (ou matrice de Gram) et où D est une matrice diagonale avec $D_i = \sum_j K(i,j)$. « weighted kernel K-means » est une version modifiée de kernel K-means (donc de K-means dans l'espace kernel) où des poids réels positifs sont donnés à chaque point (équivalent réels à des coefficients entier de multiplicité).

voir en Annexe pour plus de détails

15.6 K-means déformé

Version modifée de l'algorithme où une pondération différente est ajoutée à chaque composante de chaque point. C'est donc également une généralisation de weighted K-means. La dénomination « déformé » vient de la possibilité de généraliser encore plus en introduisant une distance de Mahanalobis (vue comme une déformation de l'espace) propre

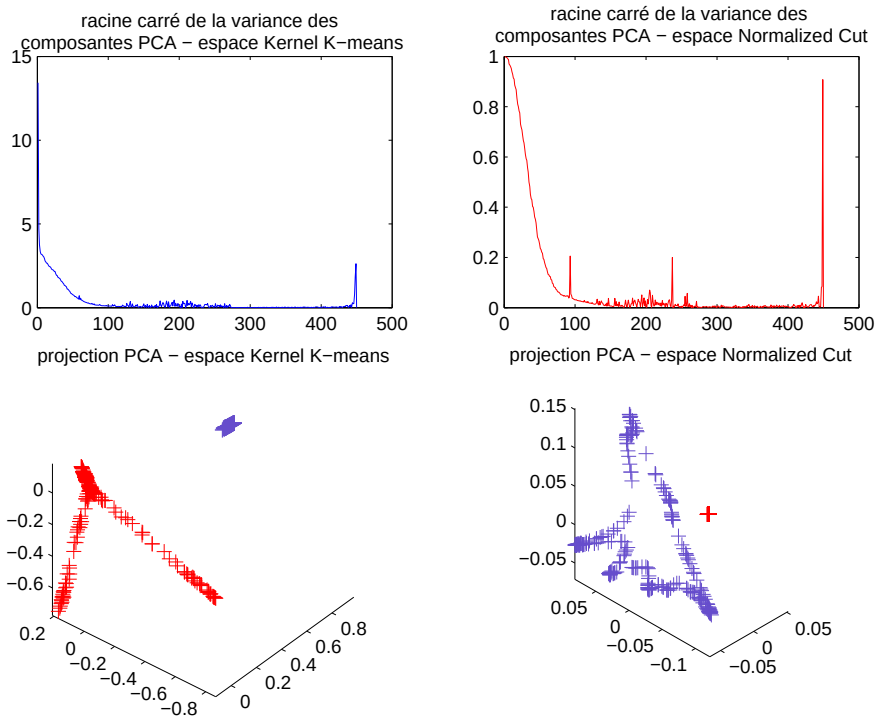

FIGURE 10 – variance PCA et projection PCA3 des points dans l'espace Kernel.

à chaque point, à l'inverse de l'algorithme K-means habituel fait uniquement usage de la distance Euclidienne.

voir en Annexe pour plus de détails

16 Det Curve comme indice de qualité d'une mesure de similarité

On définit à partir de la matrice de similarité K, un classifieur simple paramétré par un réel $\lambda \in [0;1]$.

$$C_\lambda(i,j) = \begin{cases} 1 & \text{si } K(i,j) \geq \lambda \\ 0 & \text{si } K(i,j) < \lambda \end{cases}$$

On peut alors calculer les proportions $E_P(C_\lambda)$ et $E_N(C_\lambda)$ de faux positifs et de faux négatifs donnés par le classifieur, et faire varier λ pour obtenir une Det Curve :

$$Det = \{\ (x,y) \mid x = E_P(C_\lambda), y = E_N(C_\lambda)\}$$

Le classifieur presque idéal, obtenu à partir d'une matrice de similarité proche de l'indicatrice des classes serait donc une courbe très « tassée » vers le coin $[0,0]$ en bas à gauche. Le plus mauvais classifieur, donnant des réponses aléatoires donnerait une droite oblique du point $[0,1]$ au point $[1,0]$.

Les deux courbes rouges et vertes sont assez semblables, ce qui explique également que ces deux mesures de similarités donnent à peu de chose près des résultats similaires pour le clustering. La troisième est nettement moins bonne que les deux autres, ce qui montre l'utilité d'une compression (exponentielle pour la deuxième, logarithmique pour la première).

ainsi sur une base de donnée, il vaudra mieux choisir une valeur proche de 0 alors que pour une autre une valeur proche de 1 aurait été préférable.

Fait intéressant, les MFCC habituels calculés sur l'enveloppe donnent de bien moins bons résultats que les MFC-log.

Sur ces trois figures, on peut voir que la précédence des représentations au niveau des Det Curves se vérifie aussi en clustering, ici avec l'algorithme Normalized Cut et un noyau gaussien de paramètre σ déterminé empiriquement pour chaque représentation / valeur de paramètre de compression.

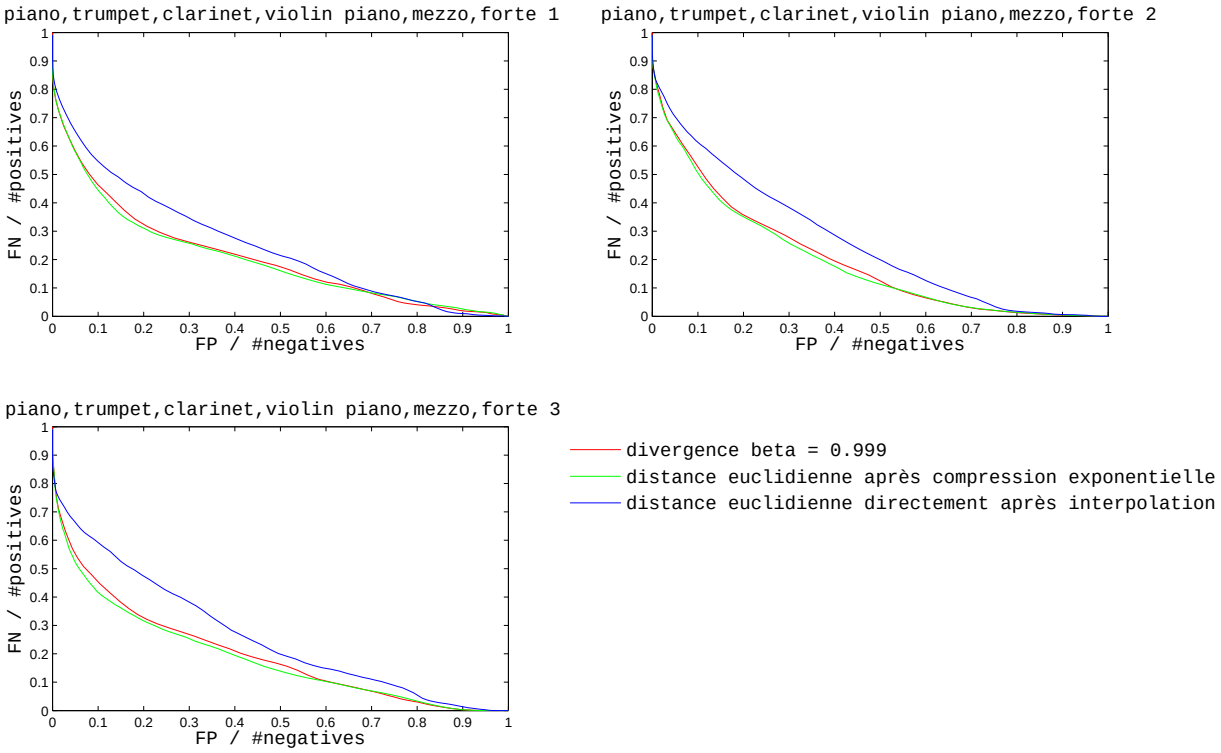

FIGURE 11 – DET Curve pour 4 instruments (clarinet piano trumpet violon), reb4 à solb4, (après l'attaque : trames 4 à 12). les trois courbes présentes représentent les résultats, pour 3 distances différentes, pour chacun des id (1,2 et 3) d'instrument de la base RWC

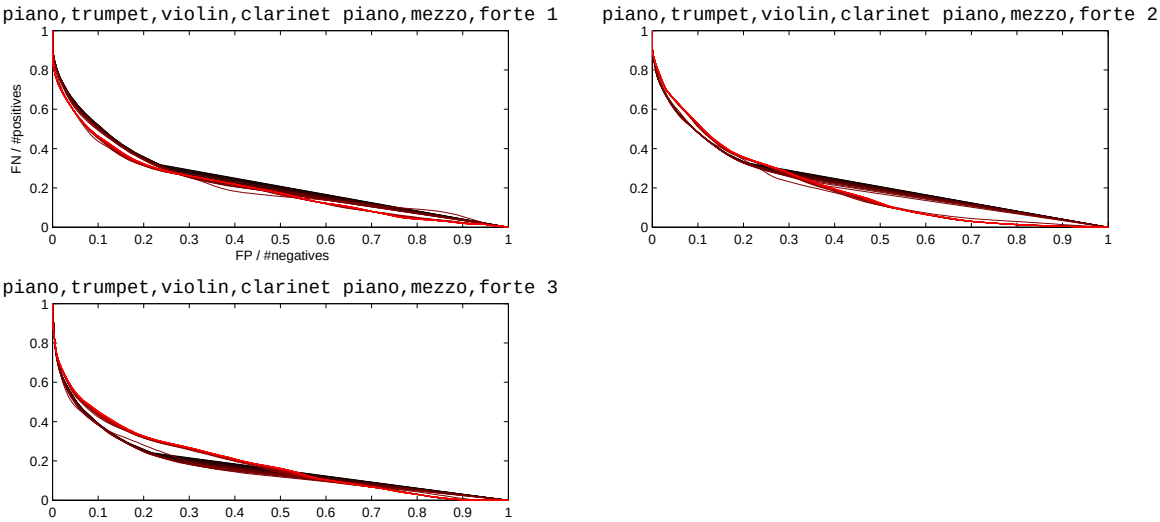

FIGURE 12 – sur cette figure on a fait varier β dans l'intervalle $]0;1[$ et on peut voir combien il est mal aisé de choisir la bonne valeur pour ce paramètre.

17 Détermination empirique des paramètres de fonction de compression

Sur la figure 16 on peut voir que les Det Curves et les performances en clustering penchant tous pour un paramètre α le plus petit possible.

Sur la figure 17 on peut voir que les Det Curves et les performances en clustering penchant tous pour un paramètre

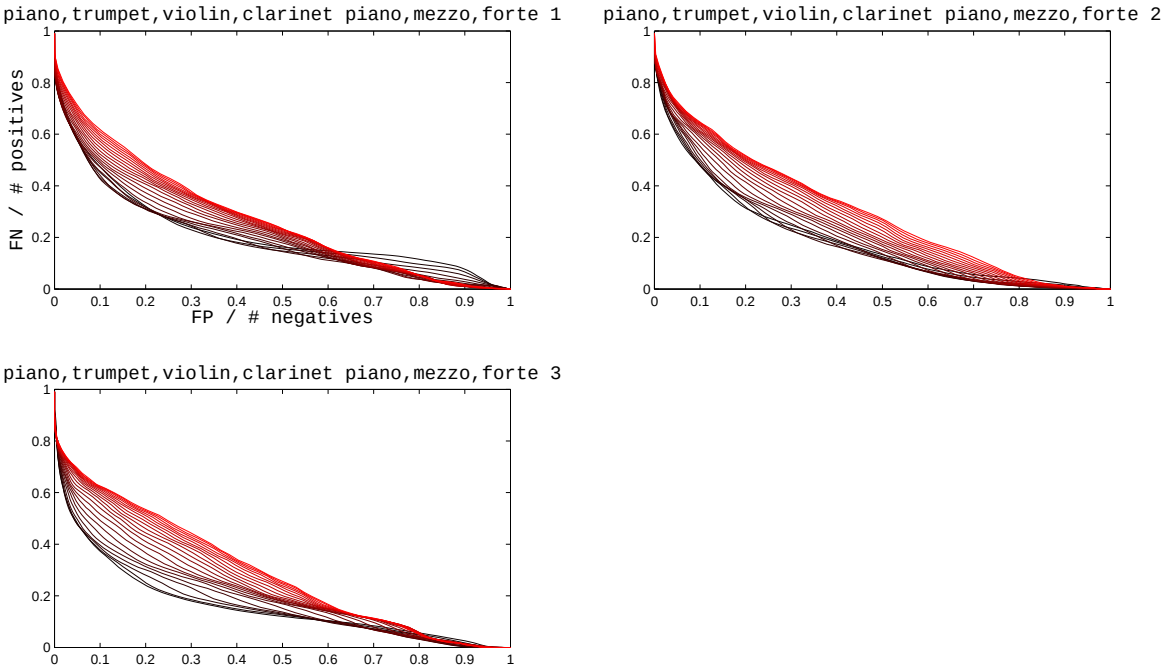

FIGURE 13 – sur cette figure on a fait varier l'exposant α dans le cadre d'une compression exponentielle et dans l'intervalle $[0.1; 1.5]$. On peut y voir qu'un facteur de compression petit (couleurs foncées, $\alpha \leq 0.4$) est préférable à un exposant supérieur à 1 (rouge vif).

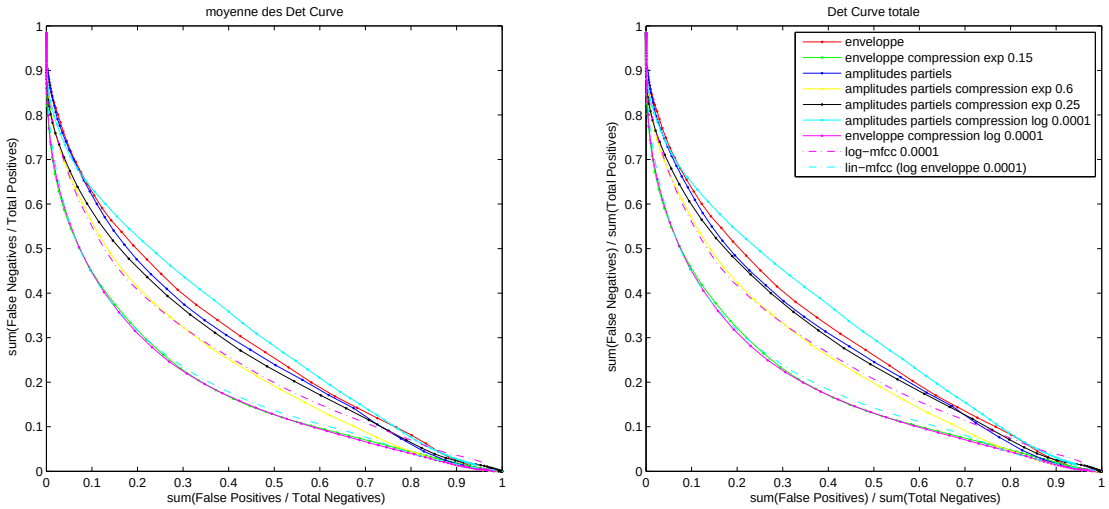

FIGURE 14 – DetCurves obtenus sur l'octave 4 pour 7 représentations différentes

α compris entre 0.08 et 0.15.

Cette méthode de validation de paramètres nous semble pertinente, et elle présente l'avantage d'être « visuelle ».

17.1 Mesures de performance d'un algorithme de clustering

En théorie, quantifier la réussite d'un clustering est très délicat. Les performances d'un algorithme de clustering pourront être très différentes sur un exemple ou un autre, et s'il est facile de dire si le résultat d'un clustering est très bon par rapport aux vraies classes, il est difficile de quantifier à partir de quel moment le résultat est considéré comme

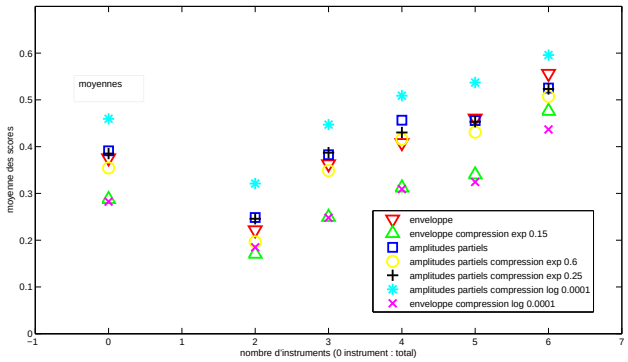

FIGURE 15 – Résultats de clustering (pourcentage d'erreur) avec l'algorithme Spectral Clustering / Normalized Cut obtenus sur l'octave 4 pour 7 représentations

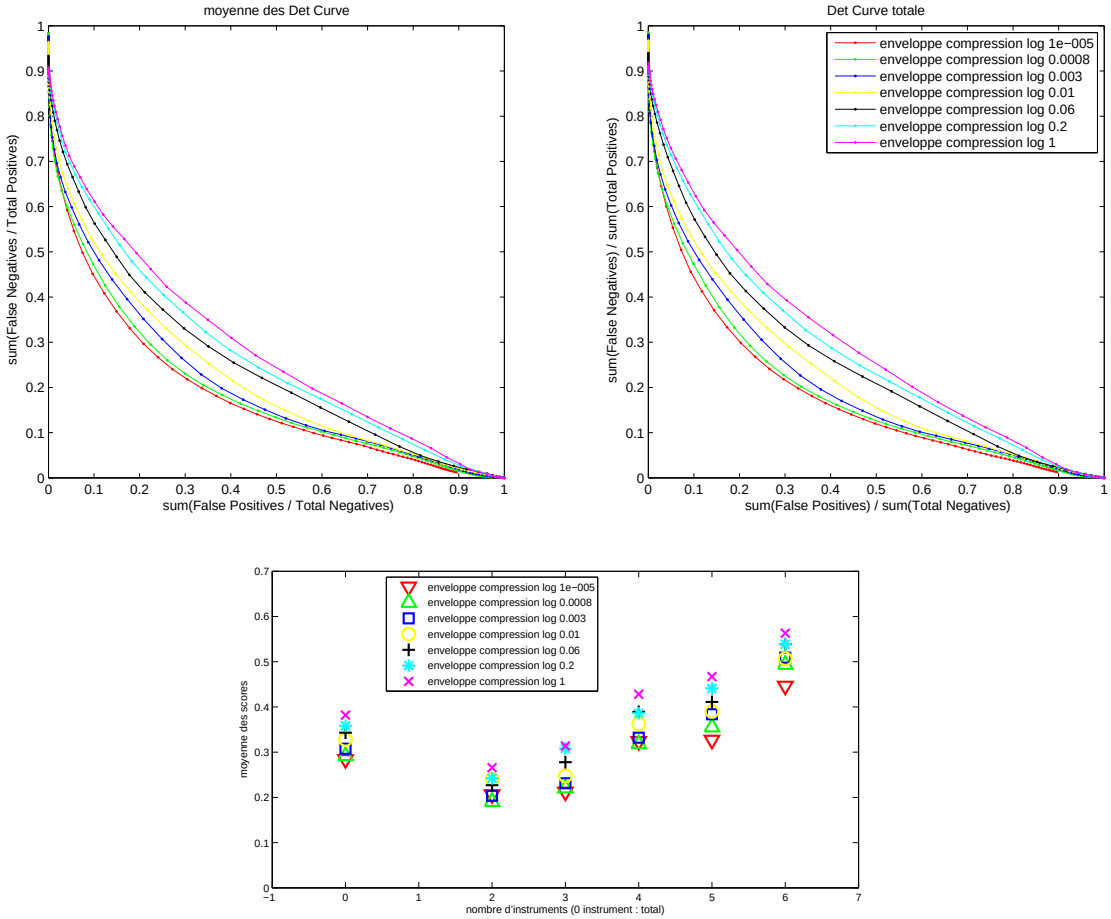

FIGURE 16 – DetCurves et résultats de clustering (pourcentage d'erreur) algorithme normalized Cut / spectral Clustering obtenus sur l'octave 4 en faisant varier le paramètre α de la compression logarithmique ($y \leftarrow \log(x + \alpha)$)

très mauvais.

Soit $[Id(1), \cdots, Id(N)] \in \{1 \cdots K\}$ les vraies classes et $[R(1), \cdots, R(N)] \in \{1 \cdots K'\}$ le résultat du clustering.

– $K = K'$, Le nombre de classes produit par le clustering est égal au nombre de classes réelles
La première mesure proposée est celle qui consiste à comparer directement le résultat du clustering avec les vraies classes.

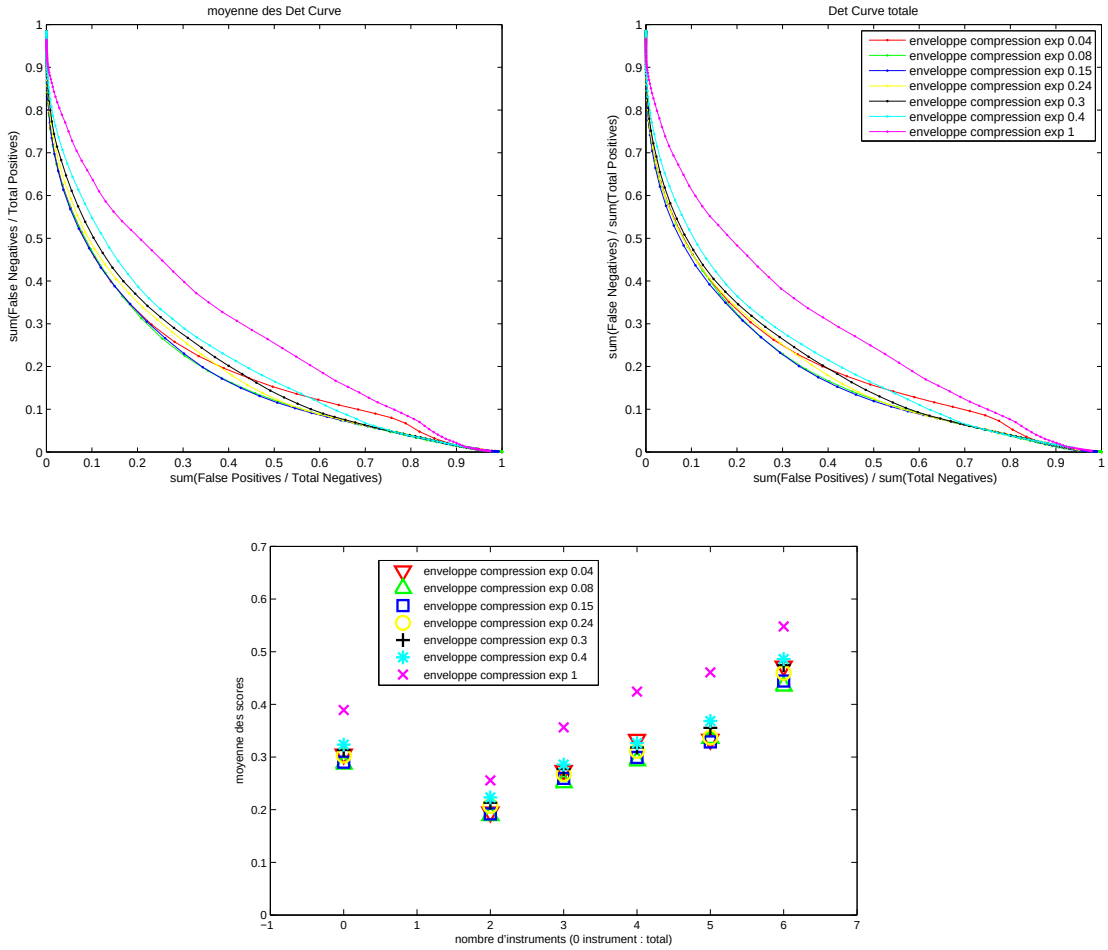

FIGURE 17 – DetCurves et résultats de clustering (pourcentage d'erreur) algorithme normalized Cut / spectral Clustering en faisant varier le paramètre α de la compression exponentielle ($y \leftarrow x^{\alpha}$)

$$M_1(R) = \min_{p \in \text{permutations}(1 \cdots K)} \sum_{n=1}^{N} (Id(n) \neq R(p(n)))$$

$M_1(R)$ représente donc le nombre minimum de déplacements d'objets, après le clustering, qu'il faudrait faire pour ranger tous les objets dans les bonnes boîtes. Cette mesure de performance (moyennée sur plusieurs exemples) est celle qui est présentée partout dans ce rapport.

– Considérer le résultat du clustering comme un classifieur binaire
Voir [II.2.1] « Det Curves»
Soit le classifieur

$$C(i,j) = \begin{cases} 1 & \text{si } R(i) = R(j) \\ 0 & \text{si } R(i) \neq R(j) \end{cases}$$

Une valeur de 1 sera interprétée comme « oui ces deux objets sont dans la même classe », alors qu'une valeur de 0 pourra soit être interprétée comme « non ces deux objets ne sont pas dans la même classe », ou alors comme « je ne sais pas ».
Notamment lorsque le nombre de classes produites par le clustering est supérieur au nombre de vraies classes, on remet en quelque sorte à plus tard la tâche de fusionner certaines de ces classes, mais on désire que les classes formées soient chacune inclues dans les vraies classes, donc qu'il n'y est pas de faux positifs.
Pour le classifieur C on peut calculer le rapport de faux positifs sur le nombre total de réponses positives :

$$E_P(C) = \frac{FP}{TN + FP}$$

Ce qui correspond au pourcentage de réponses non

et le nombre de faux négatifs sur le nombre total de réponses négatives :

$$E_N(C) = \frac{FN}{TP + FN}$$

qui n'aura donc de sens que si le nombre de classes produit par le clustering est égal au nombre de vraies classes.

- $K' > K$, le nombre de classes produit par le clustering est supérieur au nombre de classes réelles
  On peut chercher le nombre minimum de déplacements défini plus haut, si les classes produites par le clustering étaient fusionnées au mieux, ce qui donne une sorte de borne inférieure du nombre d'erreur fait par l'algorithme de clustering.

$$f : \{1 \cdots K'\} \to \{1 \cdots K\}$$
$$M_2(R) = \min_f M_1(f(R))$$

Quatrième partie
Algorithmes proposés

18 Algorithmes déformés

Dans l'article [7, MLPCA1997] on peut trouver une version plus générale et un peu différente dans son implémentation de ce que nous appelons la factorisation de matrice PCA déformée ou pondérée. Dans l'article [18, WeightedLowRankApproximations2003] la factorisation de matrice avec pondération est résolue par un algorithme itératif de type EM, plus efficace mais également plus compliqué que celui proposé ci-dessous.

Enfin, dans l'article [16, WeightedNMF2007] est proposé un algorithme NMF avec pondération équivalent à celui présenté ici.

18.1 Factorisation de matrice avec pondération

On reprend les critères utilisés par les algorithmes de factorisation de matrice K-means, PCA, NMF et Semi-NMF :

K-means
$$minJ = ||X - I\mu||^2$$
sous la contrainte que I soit une matrice indicatrice de partition

PCA
$$minJ = ||X - XPP^T||^2$$
ou de manière équivalente
$$minJ = ||X - WH||^2$$
sous la contraine que $P^TP = Id$

NMF
$$minJ = ||X - WH||^2$$
sous la contraine que W et H soient à valeurs positives

Semi-NMF
$$minJ = ||X - WH||^2$$
sous la contraine que H soit à valeurs positives

Auxquels on ajoute des poids M_{id} :

K-means déformé
$$\min J = \sum_i \sum_d (X_{id} - \sum_k I_{ik}\mu_{kd})^2 M_{id}$$
sous la contraine que I soit une matrice indicatrice de partition

PCA
$$\min J = \sum_i \sum_d (X_{id} - \sum_k W_{ik}H_{kd})^2 M_{id}$$

NMF
$$\min J = \sum_i \sum_d (X_{id} - \sum_k W_{ik}H_{kd})^2 M_{id}$$
sous la contraine que W et H soient à valeurs positives

Semi-NMF
$$\min J = \sum_i \sum_d (X_{id} - \sum_k W_{ik}H_{kd})^2 M_{id}$$
sous la contraine que H soiet à valeurs positives

voir Annexe pour les algorithmes de minimisation de chacun de ces critères (la convergence est pour chacun assurée vers un minimum local).

18.2 Application à notre problème de clustering timbral et à l'estimation de l'enveloppe

Ces algorithmes s'appliquent de manière élégante à notre problème de clustering d'instruments.

Supposons que l'on dispose de sons harmoniques respectant approximativement le modèle source-filtre : i.e une classe est définie par un filtre, et les sons dont l'on dispose sont des peignes harmoniques ainsi filtrés suivant les classes auxquelles ils appartiennent.

En réalisant l'analyse sinusoïdale d'un de ces sons, on dispose d'une évaluation du filtre de sa classe, uniquement aux fréquences multiples de la fréquence fondamentale. Notamment on ne connaît pas la forme du filtre aux fréquences plus basses que la fréquence fondamentale.

Par ailleurs, l'expérience montre que les partiels ne suivent pas parfaitement la forme du filtre et sont bruités, a forciori dans le cadre de l'estimation polyphonique.

Donc, plutôt que d'estimer la forme du filtre en réalisant une interpolation qui consiste à extrapoler les amplitudes des partiels mesurés, on propose d'utiliser tous les sons d'une classe pour estimer l'enveloppe, i.e extrapoler à partir de beaucoup plus de données, ce qui devrait donner une estimation bien meilleure.

Pour estimer les enveloppes connaissant les classes, on transforme tout d'abord notre problème.

Soit une suite d'amplitude de partiels $[a_1, a_2, \cdots, a_P]$ aux fréquences $[1f_0, 2f_0, \cdots, Pf_0]$. On fait une première estimation de l'enveloppe en utilisant par exemple l'interpolation linéaire :

$$Env(f) = \begin{cases} a_1 & \text{si } f \leq f_0 \\ a_k + \frac{f - kf_0}{f_0}(a_{k+1} - a_k) & \text{si } f \in [kf_0; (k+1)f_0] \\ a_P & \text{si } f \geq Pf_0 \end{cases}$$

Ensuite on défini une fonction de confiance en cette interpolation, on introduit :

$$w(f) = \min_p e^{-\frac{(pf_0 - f)^2}{2\sigma^2}}$$

Pour ne pas avoir à discuter de dimensions infinies, on discrétise l'axe des fréquences et on pose :

$$x_i = [Env_i(f_1) \cdots Env_i(f_d)]$$

$$\Sigma_i = \text{diag}(w_i(f_1) \cdots w_i(f_d))$$

On peut alors appliquer l'algorithme K-means déformé, ou encore PCA NMF ou Semi-NMF déformé. Les algorithmes de factorisation de matrice avec pondération pourront être appliqués car les M_{id} sont à poids positifs. Pour l'algorithme K-means déformé, Les matrices M_i étant diagonales à coefficients strictement positifs sont définies positives et les matrices $\mathcal{M}_c$ sont trivialement inversibles.

Le paramètre de régularisation σ est en quelque sorte le paramètre qui contrôle le lissage des enveloppes trouvées. Avec un axe des fréquences continu et un σ très petit, les enveloppes peuvent prendre les formes les plus accidentées, alors qu'un σ grand aura un effet de filtre passe bas.

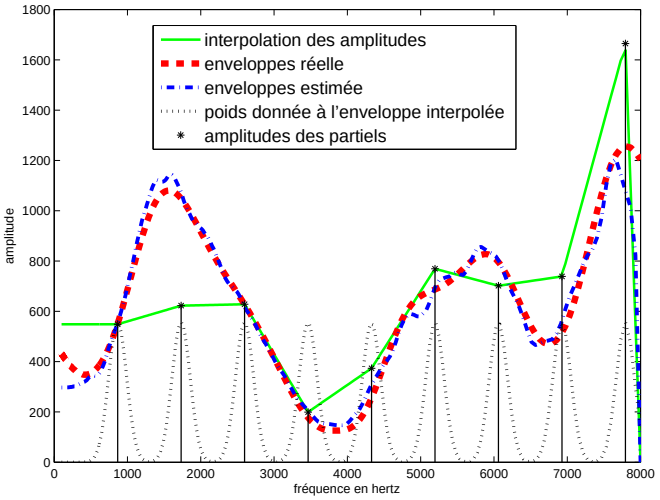

FIGURE 18 – Sur cette figure on peut voir une note synthétisée avec ses partiels régulièrement espacés en fréquence, la pondération (en noir pointillés) utilisée pour l'algorithme K-means déformé, l'enveloppe réelle à partir de laquelle la note a été synthétisée, et l'enveloppe estimée par l'algorithme K-means déformé à savoir le centroïde de la classe à laquelle cette note a été affectée.

18.3 comparaison des algorithmes pour l'estimation d'enveloppe

On propose d'évaluer ces algorithmes sur des données synthétiques. Deux ensembles de données sont synthétisés : dans le premier des notes sont synthétisées suivant le modèle de classe de filtre, alors que pour le second chaque note est synthétisée comme un mélange de plusieurs filtres de base.

Ensuite l'expérience réalisée consiste à comparer les algorithmes déformés à leurs équivalents habituels sur chacune des deux bases de données.

On remarque que plusieurs algorithmes d'estimations de l'enveloppe sont bien meilleurs que l'interpolation linéaire.

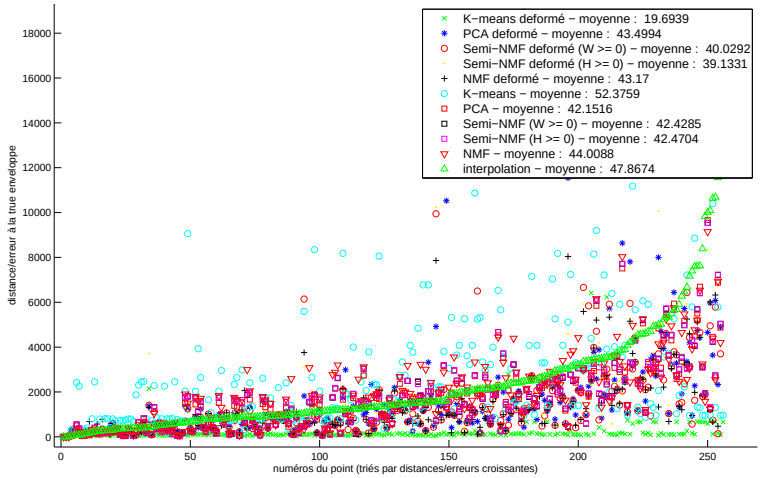

FIGURE 19 – Modèle de classes d'enveloppes. Distances entre l'enveloppe estimée et la vraie enveloppe pour chaque trame synthétisée, comparaison de 11 estimations différentes de l'enveloppe. L'erreur ou Distance moyenne sur l'ensemble des trames est donnée dans la légende.

K-means déformé donne la meilleure estimation probablement car les données suivent effectivement le modèle de classes/centroïdes.

Une autre raison importante est que K-means déformé aussi bien que K-means est très sensible à la normalisation des données, et dans notre cas les données synthétisées sont effectivement normalisées.

Il est clair que si les données/spectres synthétisés sont d'amplitudes variables, il faut chercher un moyen de les normaliser, normalisation qui sera imparfaite ce qui aura pour effet d'introduire un bruit et de réduire l'efficacité de K-means.

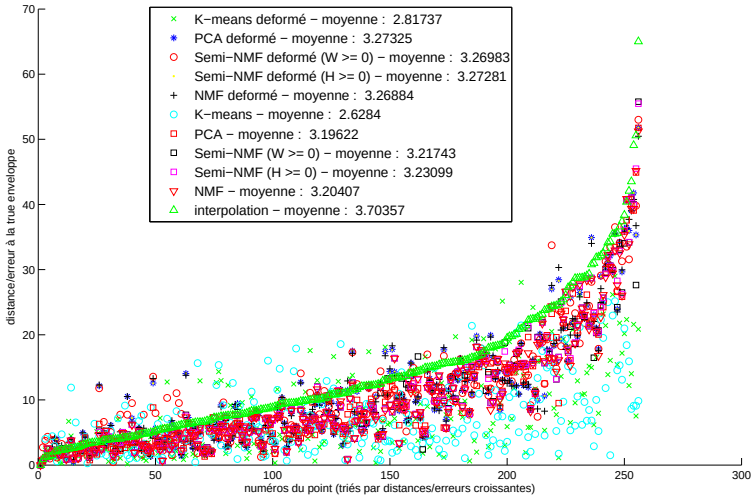

FIGURE 20 – Modèle de mélange d'enveloppes. Distances entre l'enveloppe estimée et la vraie enveloppe pour chaque trame synthétisée, comparaison de 11 estimations différentes de l'enveloppe. L'erreur ou Distance moyenne sur l'ensemble des trames est donnée dans la légende.

Ici les données ont été synthétisées avec un modèle de mélanges d'enveloppes.

C'est assez décevant, mais l'algorithme donnant les meilleurs résultats est un simple K-means sur les enveloppes interpolées. K-means déformé donne également de bons résultats, alors que PCA déformé,Semi-NMF déformé et NMF déformé donnent tous de moins bons résultats que PCA,Semi-NMF et NMF.

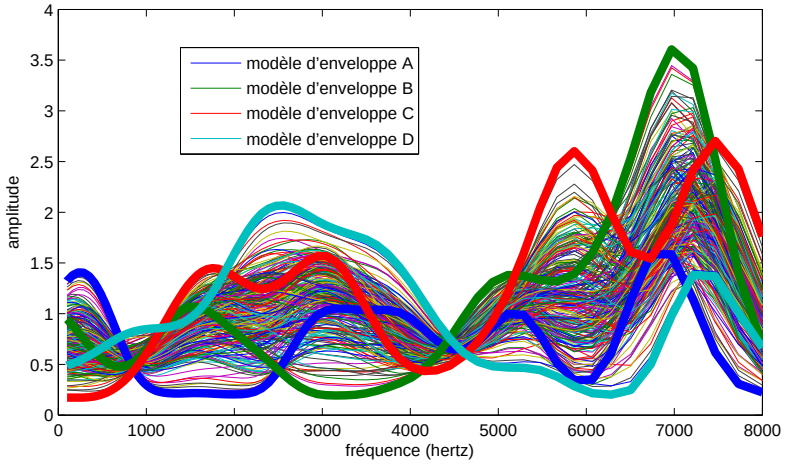

FIGURE 21 – Enveloppes obtenus après mélange de 4 enveloppes de base.

18.4 Résultats pour le problème de clustering timbral

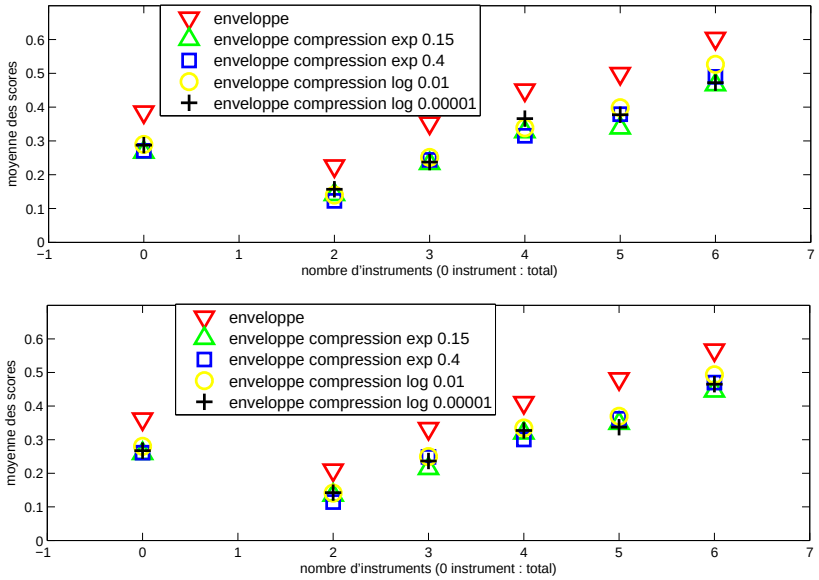

FIGURE 22 – en haut : Résultats en clustering avec l'algorithme K-means déformé pour 5 compressions d'enveloppes différentes. Base de donnée complète octave 4. en bas : Résultats en clustering avec l'algorithme K-means habituel

Sur la figure 22 on peut voir que dans le cadre de l'algorithme K-means déformé, la compression donnant les meilleurs résultats est la compression exponentielle. Les performances sont moyennes, et équivalentes à celles obtenues avec K-means normalized Cut. Les performances atteintes par un simple K-means sont meilleures! C'est donc un résultat très decevant, puisque l'algorithme le plus naïf donne des résultats aussi bons voir meilleurs que des algorithmes bien plus élaborés.

Sur la figure 23 on a renouvelé l'expérience en utilisant maintenant une base de donnée comprenant toujours 6 instruments, mais incluant en plus de l'octave 4 les octaves 3 et 5. Dans ce cas, qui se rapproche donc du cas réel où de la formation de flux / clustering en polyphonique, l'algorithme K-means déformé donne de meilleurs résultats avec une erreur moyenne de 24%. La pondération utilisée est calculée en log fréquence si bien que les hautes fréquences reçoivent une pondération très proche de 1 alors que seules les fréquences basses très proches d'un partiel on une valeur significativement non nulle. Il serait intéressant d'évaluer d'autres pondérations, celle-ci ayant visiblement tendance à sur-pondérer les hautes fréquences, par exemple en divisant les pondérations par leur moyenne locale pour que celle-ci soit toujours égale à 1.

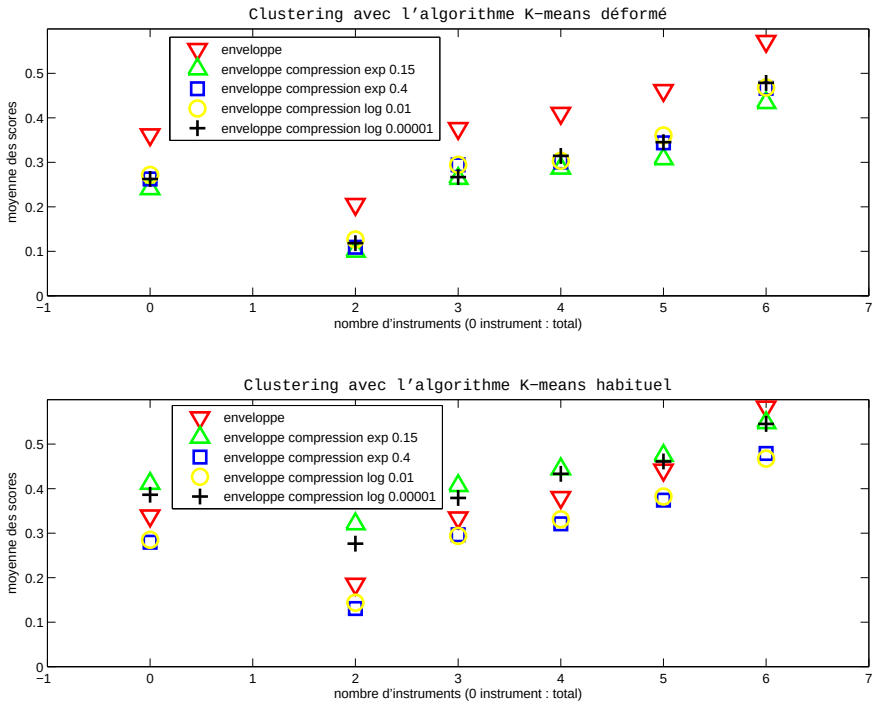

FIGURE 23 – la Base de donnée comprend ici 3 octaves (Do3 à Si5). en haut : Résultats en clustering avec l'algorithme K-means déformé pour 5 compressions d'enveloppes différentes. en bas : Résultats en clustering avec l'algorithme K-means habituel

‘

19 Algorithmes K-means dans l'espace source-filtre

19.1 Problème de la régularisation

19.1.1 régularisation de Tikhonov

dans le cas général le problème de régression suivant est sous déterminé :

$$\mu = \underset{\mu}{\operatorname{argmin}} ||A - \mu B||^2$$

en effet on trouve comme condition pour que μ soit optimal :

$$\mu B B^T = A B^T$$

si B n'est pas de rang plein, BB^T n'est pas inversible et il existe une infinité de solutions.
Il est alors nécessaire de régulariser le problème, par exemple à l'aide de la régularisation de Tikhonov [10] :

$$\mu = \underset{\mu}{\operatorname{argmin}} ||A - \mu B||^2 + \alpha ||\mu||^2$$

avec $\alpha > 0$. Ce qui donne lieu à l'équation suivante :

$$\mu(BB^T + \alpha I) = AB^T$$

comme $(BB^T + \alpha I)$ est toujours inversible, la solution est unique et le problème peut être résolu numériquement.

10. il est possible de remplacer la norme euclidienne par n'importe quelle norme. Pour que le problème reste facile à résoudre on choisit en général : $||\mu||^2_\Sigma = \mu \Sigma \mu^T$, la condition nécessaire pour que le problème admette une unique solution étant que la matrice Σ soit définie positive.

19.1.2 à l'aide d'un programme quadratique

soit le programme quadratique :

$$\begin{aligned} & \min ||\mu||^2 \\ s.c \quad & \mu BB^T = AB^T \end{aligned}$$

en utilisant la décomposition en valeurs singulières $B = USV^T$ on a :

$$\begin{aligned} & \min ||\mu||^2 \\ s.c \quad & \mu USS^TU^T = AVS^TU^T \end{aligned}$$

ou encore

$$\begin{aligned} & \min ||\mu U||^2 = ||x||^2 \\ s.c \quad & \mu USS^T = AVS^T = xSS^T \end{aligned}$$

la solution est alors de choisir l'unique $x \in vec(S)$ satisfaisant la contrainte et de norme minimale, que l'on trouve à l'aide de la matrice diagonale : $T(i,i) = 1/S(i,i)^2$ si $S(i,i) \neq 0$, 0 sinon.

Cette méthode permet de ne pas utiliser de terme de régularisation, qui introduit un biais dans la valeur du critère. Néanmoins, nous ne l'utilisons pas pour deux raisons :

1. cette méthode est couteuse en calculs et nécessite une décomposition spectrale de la matrice
2. le terme de régularisation est équivalent dans la plupart des algorithmes considérés dans ce rapport à une « contrainte de lissage », et devient ainsi indispensable.

19.2 Interpolation matricielle des partiels

Soit x_i une suite d'amplitudes de partiels, e l'enveloppe définie sur n points et Π_i la matrice encodant les fréquences des partiels tel que :

$$x_i \simeq ePi_i$$

Si l'on souhaite réaliser une interpolation basse fréquence de x_i on peut calculer e comme suit :

$$e = x_i\Pi_i^T H(H\Pi_i\Pi_i^T H + \lambda I)^{-1}H$$

où H est une matrice Toeplitz de filtrage passe bas, et λ une constante réelle proche de 0.

Si l'on souhaite réaliser une interpolation linéaire de x_i on peut calculer e comme suit :

$$e = x_i\Pi_i^T(\Pi_i\Pi_i^T + \lambda L)^{-1}$$

où $L : n \times n$ tel que : $L(n,n) = 1$, pour toutes les colonnes sauf la première et la dernière : $L(n-1,n) = L(n,n+1) = -1/2$, $L(2,1) = L(n-1,n) = -1$. Il est clair que si e est une interpolation linéaire comme sur le graphique, eL est nul partout sauf aux fréquences des partiels. En minimisant $J = ||x_i - e\Pi_i||^2 + \lambda eLe^T$ on trouve la formule donnée au dessus. Nous passons sous silence le fait que L n'est pas définie positive, et les problèmes d'inversibilité de la matrice $\Pi_i\Pi_i^T + \lambda L$.

Sur la figure 24 on voit graphiquement le résultat obtenu pour chacune des interpolations.

Enfin sur la figure 25 on peut voir que les Det Curves obtenues avec des enveloppes otenues par interpolation basse fréquence (LF sur le graphique) sont un peu moins bonnes que celles obtenues par interpolation linéaire.

19.3 K-means dans l'espace des enveloppes

19.3.1 Pourquoi les centroïdes devraient être des enveloppes

Dans [2, Burred2009] il est montré qu'après interpolation des amplitudes de partiels, la projection de dimension J donnée par l'analyse en composante principale (PCA) engrendre une erreur plus faible que si la même opération est appliquée aux amplitudes de partiels directement.

Plus formellement :

Soit X une matrice contenant sur chaque ligne des amplitudes de partiels calculées sur des échantillons sonores de plusieurs instruments. Soit Y la matrice contenant sur chaque les enveloppes obtenues par interpolation des amplitudes

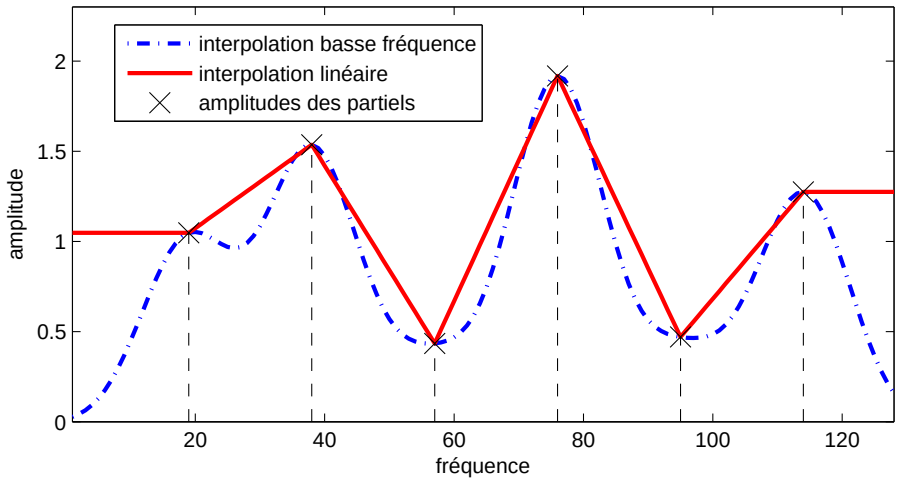

FIGURE 24 – Exemple d'enveloppe obtenue par interpolation linéaire et interpolation basse fréquence d'une suite d'amplitudes de partiels

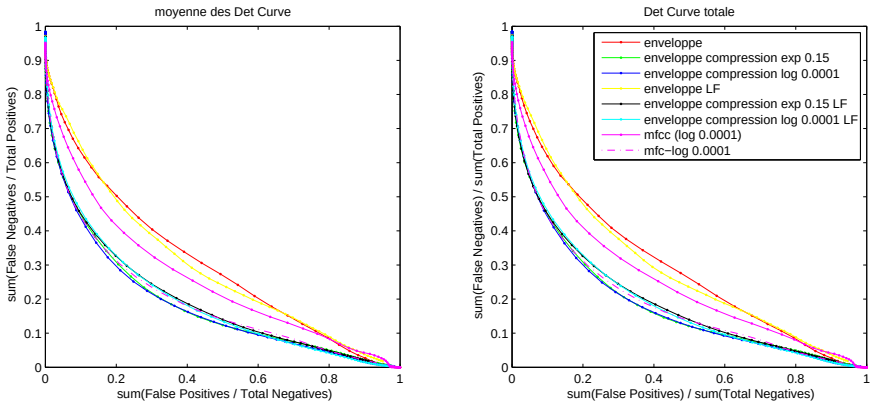

FIGURE 25 – Det Curves obtenues sur la base de donnée comprenant l'octave 4, montrant que l'interpolation linéaire donne de meilleurs résultats que l'inteprolation basse fréquence (LF dans la légende).

de partiels contenues dans X. Soit f la tranformation permettant de retrouver à partir de Y les amplitudes des partiels (pour cela la transformation f connait donc les fréquences de chaque partiel) : $X \simeq f(Y)$.

Soit P_X la matrice des K premiers vecteurs propres de $X^T X$ (K varie entre 6 et 10 dans [2, Burred2009]) et P_Y la matrice des K premiers vecteurs propres de $Y^T Y$.

Sur la base de donnée utilisée dans [2, Burred2009] on a :

$$||X - XP_X P_X^T|| > ||X - f(YP_Y P_Y^T)||$$

Il est ainsi montré que les « features » dépendant de la fréquence sont plus déterminantes que les « features » dépendant du numéros de partiel. Cela nous amène à proposer une version modifée de l'algorithme K-means où les points à grouper restent des amplitudes de partiels, mais où les centroïdes évoluent eux dans l'espace des envelopes.

19.3.2 algorithme K-means dans l'espace des enveloppes

Pour chaque couple (x_i, y_i) d'enveloppe interpolée et d'amplitudes de partiels, il existe une matrice binaire M_i encodant les fréquences des partiels telle que $x_i \simeq y_i$. Soit un ensemble d'amplitudes de partiels appartenant à une même classe, regroupées autour d'un même centroïde. On définit alors le centroïde comme étant le vecteur μ minimisant :

$$J_e = \sum_i ||x_i - \mu M_i||^2$$

ce critère pose un problème car rien ne nous assure que le centroïde, supposé être une enveloppe modélisant bien les x_i soit lisse en fréquence. Pour cela, on remplace μ par eH, où H est une matrice Toeplitz de filtrage passe-bas ($H_{ij} = h(i-j)$ où h est la réponse impulsionnelle d'un filtre passe bas, par exemple une fenêtre de Blackman).

$$J_e = \sum_i ||x_i - eHM_i||^2$$

en annulant le gradient du critère on trouve alors la relation suivante :

$$eH(\sum_i M_i M_i^T)H = \sum_i x_i M_i^T$$

Cette équation peut être sous-déterminée. Par ailleurs, si la matrice H est inversible ou même « presque inversible » (peu de ses valeurs propres sont nulles), son rôle de filtrage passe bas n'est plus effectif. Pour résoudre ces deux problèmes, on ajoute un terme de régularisation, et on obtient :

$$J_e = \sum_i ||x_i - eHM_i||^2 + \gamma ||e||^2$$

$$e = \sum_i x_i M_i^T (H(\sum_i M_i M_i^T)H + \gamma I)^{-1}$$

Maintenant qu'à partir d'un ensemble d'amplitudes de partiels on sait trouver l'enveloppe ou le centroïde qui les explique au mieux, on peut adapter l'algorithme K-means :

$$J = \sum_c \sum_{i \in c} ||x_i - e_c H M_i||^2 + \gamma \sum_c ||e_c||^2$$

$$e_c = (\sum_{i \in c} x_i M_i^T)(H \sum_{i \in c} M_i M_i^T H + \gamma I)^{-1}$$

La convergence de cette algorithme est montrée dans le paragraphe 24.

19.3.3 Réduction de la dimension des vecteurs d'enveloppe

La dimension des e_c peut être fortement réduite :
la SVD de la matrice H est calculée :

$$H = USV^T$$

En retirant de U, S, V les valeurs singulières très proches de 0 et leurs vecteurs singuliers associés (qui forment le noyau ou « null space» de H), on obtient une bonne approximation de H

$$H \simeq \tilde{H}$$

$$eH \simeq e\tilde{U}\tilde{S}\tilde{V}^T$$

$$eH \simeq \tilde{e}\tilde{S}\tilde{V}^T$$

où la dimension de $\tilde{e}$ est fortement réduite par rapport à celle de e. La régularisation n'est pas affectée par cette transformation puisqu'on a également :

$$||e||^2 \simeq ||\tilde{e}||^2$$

Dans la pratique, si on choisit d'échantilloner les enveloppes sur 256 points et d'utiliser comme filtre une fenêtre de Blackman de 43 points, on peut avec une erreur inférieure à 0.1% diminuer la dimension à 60, équivalent d'un sous-échantillonnage d'un facteur 4 par analogie avec le traitement du signal.

19.4 K-means avec modèle de source multiplicative

On souhaite introduire un modèle de source (ou modèle de pondération des partiels : voir 5) sous la forme d'une pondération des partiels dépendante de la source, on propose d'ajouter au le critère défini dans le paragraphe 19.3 précédent, pour chaque classe, une matrice diagonale $D_c = \text{diag}([d_{c1} \cdots d_{cP}])$ encodant la pondération dépendant uniquement de la source (donc du numéros de partiel) :

$$J_3 = \sum_c \sum_{i \in c} ||x_i - e_c H M_i D_c||^2$$

où e_c est une enveloppe possiblement chaotique, $e_c H$ est une enveloppe lissée, $e_c H M_i$ est la projection de l'enveloppe aux fréquences des partiels de l'observation i, D_c est la pondération des partiels dépendante de la source, et les x_i sont les suites d'amplitudes de partiels à expliquer.

Cette expression pose un problème : il manque un terme de régularisation, et la multiplication/division par une constante de e_c et D_c ne change pas la valeur du critère.

En plus des termes de régularistion, on ajoute également une contrainte de positivité : $D_c \geq a$ avec $a > 0$. (par la suite on montre que la valeur de a qui donne les meilleurs résultats pour la normalisation des x_i utilisée est $a = 2$).

L'algorithme K-means modifié est le suivant :

1. initialiser aléatoirement les partitions, $D_c = \text{diag}([a \cdots a])$.
2. optimiser les e_c connaissant les partitions et les D_c avec comme paramètre de régularisation $\gamma > 0$

$$e_c = \sum_i x_i D_c M_i^T (H(\sum_i M_i D_c^2 M_i^T) H + \gamma I)^{-1}$$

3. optimiser [11] les D_c connaissant les partitions et les e_c avec comme paramètre de régularisation $\delta > 0$ et sous la contrainte que $d_{cp} > a$

$$d_c(p) = \max(a, (\sum_{i \in c} e_c H M_i(.,p) x_i(p)) / (\sum_{i \in c} (e_c H M_i(.,p))^2 + \delta))$$

4. recalculer les partitions avec la méthode habituelle
5. retourner à l'étape 2 jusqu'à convergence

On propose également un algorithme GEM (generalized EM) après avoir introduit des probabilités gaussiennes et modèle probabilite type « gaussian mixture model ».

19.5 K-means avec modèle de source additive

Avec le modèle de source multiplicative, l'amplitude du partiel numéro p d'une note d'un certain instrument de fréquence fondamentale f_0 on écrit que :

$$a_p \simeq A d_p e(p f_0) + \epsilon$$

où A est l'amplitude de la note indépendante du numéro de partiel.

En passant au logarithme, on arrive à un modèle de source additive qui s'écrira :

$$\log a_p = \log A + \log d_p + \log e(p f_0) + \epsilon$$

cela peut s'écrire de manière matricielle : (en n'écrivant pas les log pour alléger l'écriture et avec x_i le vecteur des amplitudes de partiels d'une certaine note)

$$x_i = A_i + \mu \Pi_i + \vec{\epsilon}$$

où $\mu = [d, e]$ est le centroïde de la classe à laquelle appartient la note i. Ce centroïde définit une pondération dépendante du numéros de partiel et une pondération dépendante de la fréquence. En log celle-ci devient un terme additif.

La partie supérieure de la matrice Π_i correspond à la matrice identité et la partie inférieure à HM_i du paragraphe précédant.

En passant au logarithme, deux modifications importantes on été réalisées :

11. on peut écrire $J(D) = \sum_c \sum_p J_{cp}(d_c^p)$ si bien qu'on se retrouve avec $P \times C$ fonctions quadratiques d'une variable à minimiser avec contrainte. Chacune de ces fonctions est du type $f(x) = \alpha x^2 + \beta x + \gamma$ avec $\alpha > 0$ et sous la contrainte que $x \geq a \geq 0$. Le minimum est alors atteint pour $x = \max(a, \beta/(2\alpha))$

- si le bruit additif ϵ était supposé centré en linéaire, en passant au logarithme il ne sera plus, et inversement. Le bruit additif est donc modélisé autrement, et la minimisation des moindre carrés aura un sens différent.
- la matrice H assure que l'enveloppe soit lisse, cependant il est différent d'imposer que l'enveloppe soit lisse dans le domaine linéaire ou dans le domaine logarithmique.

Ces précisions apportées, rien ne nous empêche de proposer et d'étudier les performances d'un algorithme de classification et une version modifiée de K-means intégrant ce modèle. Le critère permettant de calculer un centroïde sera :

$$J = \sum_i ||x_i - \mu\Pi_i - A_i||^2 + \gamma||\mu||^2$$

Si les x_i sont centrés ($\sum_p x_{ip} = 0$) alors on aura $A = \sum_p (\mu\Pi_i)_p / P$, si bien que notre critère sera équivalent à :

$$J = \sum_i ||x_i - \mu\Pi_i(I - 1/P)||^2 + \gamma||\mu||^2$$

la matrice $I - 1/P$ permettant de centrer le vecteur $\mu\Pi_i$. Avec $\tilde{\Pi}_i = \Pi_i(I - 1/P)$ et en annulant allant le gradient on trouve :

$$\mu = (\sum_i x_i \tilde{\Pi}_i^T (\sum_i \tilde{\Pi}_i \tilde{\Pi}_i^T + \gamma I)^{-1}$$

Et enfin on peut proposer l'algorithme de type K-means :

1. initialiser aléatoirement les partitions.
2. optimiser les μ_c connaissant les partitions

$$\mu_c = (\sum_{i \in c} x_i \tilde{\Pi}_i^T)(\sum_{i \in c} \tilde{\Pi}_i \tilde{\Pi}_i^T + \gamma I)^{-1}$$

3. recalculer les partitions avec la méthode habituelle
4. retourner à l'étape 2 jusqu'à convergence

19.6 Autre linéarisation du modèle

La modèle de source multiplicative s'écrit :

$$a_p \simeq Ae(pf_0)d_p + \epsilon$$

en posant $w_p = 1/d_p$ on obtient :

$$a_p w_p \simeq Ae(pf_0) + \tilde{\epsilon}$$

On peut donc écrire un critère à minimiser pour estimer les paramètres du modèle :

$$J = \sum_i ||wX_i - e\Pi_i||^2$$

Ou en reprenant la démarche déjà présentée :

$$J_R = \sum_i ||wX_i - eH\Pi_i||^2 + \alpha||e||^2 + \beta||w||^2$$

Il manque évidemment des contraintes sur w et e, car ce critère sans contrainte est trivialement minimisé pour $w = 0, e = 0$. Nous avons réalisé plusieurs essais :

1. $||w||^2 = 1$
2. $||e||^2 = 1$
3. $w \geq 1$
4. $e \geq 1$

Les deux premières contraintes donnent lieu à un calcul de vecteur propre, les deux secondes à un programme quadratique avec matrice définie positive donc efficament résolu.

Nous n'avons pas obtenu de réel succès puisque dans chaque cas, les performances en classification supervisée étaient moins bonnes qu'avec un centroïde estimé comme la moyenne des vecteurs d'amplitudes.

Nous avons également essayé un modélie inversé :

$$a_p v_(pf_0) \simeq Ad_p + \tilde{\epsilon}$$

où d_p est la pondération du partiel p, et $v_(pf_0) = 1/e(pf_0)$. Cette modélisation est différente en terme de bruit additif mais amène aux même outils de calcul lieu aux mêmes calculs et ne donne pas de meilleurs résultats.

19.7 Le problème de la normalisation de l'amplitude

Prenons par exemple le modèle de source multiplicative :

$$x_i \simeq e_c H M_i D_c$$

$$J = \sum_c \sum_{i \in c} ||x_i - e_c H M_i D_c||^2$$

Le critère n'est pas invariant par multiplication des x_i par des constantes différentes.

C'est pourquoi on commence par normaliser les x_i. Cependant, cette démarche n'est pas consistante avec le modèle, il serait plus logique d'ajouter un réel $a_i > 0$ pour chaque i :

$$x_i \simeq a_i e_c H M_i D_c$$

$$J = \sum_c \sum_{i \in c} ||x_i - a_i e_c H M_i D_c||^2$$

on normalise toujours les x_i car cela diminue les chances que les a_i soient très différents, à l'optimum. On peut toujours estimer les paramètres du modèle avec un algorithme de type « alternating least square ». Cependant, la complexité du modèle est d'un ordre plus élevé (d'ordre 6 au lieu de 4), et cela augmente probablement le nombre de minimum locaux. Toujours est-il que nous avons utilisé ce modèle, puis nous l'avons abandonné car il avait l'air de donner de moins bonnes performances. Il serait donc intéressant de vérifier rigoureusement l'influence de l'utilisation ou non d'un terme de normalisation intégré au modèle, en classification et clustering.

Pour le modèle additif, la situation est plus simple puisque le modèle reste quadratique, et peut même éliminer les a_i qui dépendent linéairement des autres paramètres à l'optimum.

19.8 L'évolution des paramètres de régularisation

Prenons par exemple le modèle de source multiplicative avec les termes de régularisation :

$$J = \sum_c \sum_{i \in c} ||x_i - e_c H M_i D_c||^2 + \gamma ||e||^2 + \delta ||D||^2$$

Lorsque γ devient très grand, J est équivalent au critère des K-means. A l'inverse lorsque δ devient très grand J est équivalent au critère K-means dans l'espace des enveloppes (paragraphe 19.3).

Qu'en est-il si les paramètres de régularisation évoluent au cours des itérations pour l'estimation des paramètres du modèle ? Ceux-ci introduisent un biais, et donc les faire diminuer au cours des itérations n'est pas illogique. Dans plusieurs articles sur NMF il est montré une augmentation des performances en appliquant cette stratégie. Nous n'avons pas essayé pour notre problème et il serait probablement intéressant de le faire.

19.9 Résultats pour notre problème de clustering

Sur la figure 26 on peut voir que les algorithmes de clustering donnent tous des performances assez similaires, toutefois il semble que la complexité croissante (de gauche à droite, l'algorithme K-means étant le plus plus naïf) amène également des performances croissantes. C'est donc en quelque sorte un succès qui justifie l'étude de modèles plus complexes.

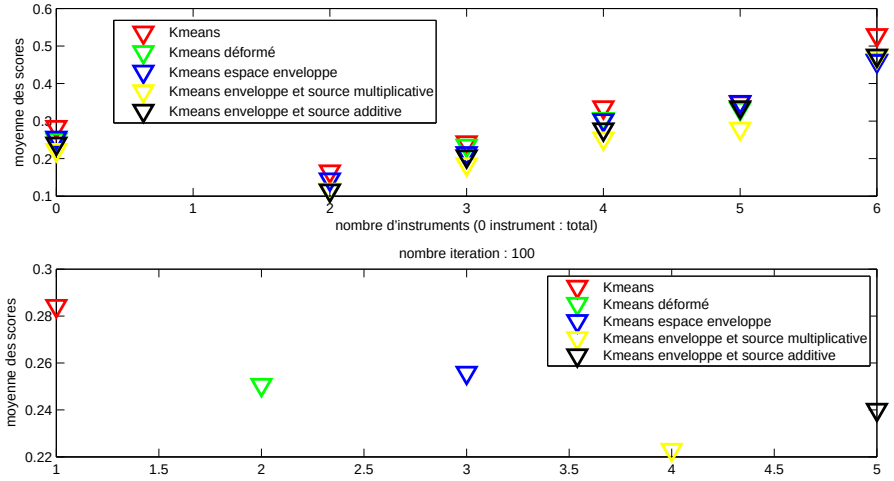

FIGURE 26 – Comparaison de 5 algorithmes, résultats (pourcentage d'erreur) en clustering sur la base de donnée comprenant 3 octaves. Le plus mauvais résultat est donné par un K-means naïf sur les amplitudes partiels compressées.

20 Pondération optimale des Features dans l'espace de description

L'idée sous jacente est la suivante : lorsque l'on calcule beaucoup de descripteurs à partir de signaux ou d'enveloppes dans le but de les utiliser en reconnaissance d'instrument ou pour réaliser un clustering, on doit d'une part choisir quelles sont les features les plus significatives, et d'autre part affecter un poids à chacune des features en espérant qu'ainsi, le calcul d'une distance ou d'une mesure de similarité (distance euclidienne ou application d'un noyau gaussien) aux vecteurs de features pondérés de chacune des notes ou trames observées soit significative.

Cette idée généralise le concept de sélection de descripteurs (« features selection ») souvent utilisé en reconnaissance d'instrument. La sélection de descripteurs consiste à affecter un poids 0 ou 1 à chaque descripteur, les descripteurs affectés d'un poids nul étant donc non utilisés. Cette méthode est utilisée par exemple dans [3, Eronen2000].

Une autre méthode utilisée également en reconnaissance d'instrument consiste à procéder à une analyse linéaire discriminante (LDA, voir paragraphe 28), donc à calculer un ensemble réduit de descripteurs abstraits, chacun étant correspondant à une combinaison linéaire des descripteurs initiaux. Cette méthode est utilisée par exemple dans [14, Kitahara2003].

Une dernière méthode utilisée dans [17, Kinoshita2006] consiste à pondérer les descripteurs en fonction de la « confiance » que l'on peut avoir en chacun d'eux lorsqu'ils sont extraits d'un mélange polyphonique. Les descripteurs considérés sont supposés être fortement discriminant lorsqu'ils sont extraits de signaux monophoniques.

Ici nous proposons une méthode différente, qui consiste à ne pas considérer de combinaison linéaire des descripteurs mais à affecter chaque descripteur d'un poids réel positif, poids automatiquement estimés pour optimiser la bonne répartition des points dans l'espace kernel induit par le noyau gaussien.

On propose donc un algorithme supervisé pour optimiser cette pondération.

soit $X = [x_1 \cdots x_N]^T$ une matrice d'observations, où chaque ligne contient un vecteur de feature. $L = [L_1 \cdots L_N] \in C^N$ les classes auxquelles appartiennent les observations, et C le nombre de classes

soit $Y = [y_1 \cdots y_N]^T$ la matrice des observations après transformation par un kernel gaussien (voir III.3.1 pour une formulation explicite)

On cherche à modéliser chacune des classes dans cet espace par une boule (ce qui est en accord avec l'intention d'utiliser les kmeans pour réaliser un clustering ou une classification dans cet espace).

20.1 critère proposés

20.1.1 premier critère

Ainsi un critère possible est de minimiser la distance de chaque point au centroïde de la classe à laquelle il appartient tout en maximisant la distance aux centroïdes des autres classes.

soit $A : N \times C$,

$$A_{ic} = \begin{cases} (C-1)/C & \text{si } L(i) = c \\ -1/C & \text{si } L(i) \neq c \end{cases}$$

pour un point x_i on cherche à minimiser :

$$\sum_{c=1}^{C} A_{ic} ||y_i - \mu_c||^2$$

où μ_c est le centroïde de la classe c

les valeurs données aux coefficient de A expriment le fait que l'on cherche à minimiser la distance au carré de y_i au centroïde de sa classe moins la moyenne des distances au carré de y_i aux centroïdes des autres classes

deux critères sur l'ensemble des points sont proposés :

$$J_1 = \sum_{i=1}^{N} \sum_{c=1}^{C} A_{ic} ||y_i - \mu_c||^2$$

$$J_2 = \sum_{i=1}^{N} tanh(\gamma + \beta \sum_{c=1}^{C} A_{ic} ||y_i - \mu_c||^2)$$

l'ajout d'une fonction *tanh* modélise le fait que si un point est « suffisamment » proche du centroïde de sa classe par rapport aux centroïdes des autres classes, le critère est considéré comme satisfait et on ne cherche pas à l'éloigner plus que nécessaire des centroïdes des autres classes. à l'inverse, si un point est un outlier ou est en fait mal classé, la pénalité est majorée par la fonction tanh indépendament des autres points.
pour cette raison le second critère donne de bien meilleurs résultats que le premier, spécialement dans les cas difficiles

20.1.2 second critère (Fisher criterion)

Le critère est cette fois plus connu : minimiser la variance intra classe tout en maximisant la variance inter classe (connu comme le « Fisher Discriminant Criterion »).

$$J_3 = \frac{\sum_{c=1}^{C} \sum_{\substack{i=1 \\ L(i)=c}}^{N} ||y_i - \mu_c||^2}{\sum_{c=1}^{C} \sum_{c'=1}^{C} ||\mu_c - \mu'_c||^2}$$

Ce critère présente le défaut selon nous de ne pas traiter différemment les outliers des autres, ce que nous tentons de faire en introduisons une fonction seuil telle que la fonction *tanh*.

20.1.3 troisième critère (critère quotient)

On peut représenter l'affinité d'un point pour une classe par rapport à une autre comme

$$Aff(y_i, n, m) = \log \frac{||y_i - \mu_m||^2}{||y_i - \mu_n||^2}$$

cette affinité peut être arbitrairement grande ou faible, et pour limiter le poids donné aux points très proches d'un centroïde ou à ceux au contraire très éloignés, on peut ajouter une fonction tanh, l'affinité devient alors :

$$Aff(y_i, n, m) = \tanh\left(\beta \log \frac{||y_i - \mu_m||^2}{||y_i - \mu'_n||^2}\right)$$

comme $\tanh(u) = \frac{e^{2u}-1}{e^{2u}+1}$ on trouve :

$$Aff(y_i, n, m) = \tanh\left(\log\left(\frac{||y_i - \mu_m||^2}{||y_i - \mu_n||^2}\right)^\beta\right) = \frac{\left(\frac{||y_i-\mu_m||^2}{||y_i-\mu_n||^2}\right)^\beta - 1}{\left(\frac{||y_i-\mu_m||^2}{||y_i-\mu'_n||^2}\right)^\beta + 1} = \frac{(||y_i - \mu_m||^2)^\beta - (||y_i - \mu_n||^2)^\beta}{(||y_i - \mu_m||^2)^\beta + (||y_i - \mu_n||^2)^\beta}$$

un critère de bonne repartition des points sera alors :

$$J = \sum_{i=1}^{N}\sum_{c=1}^{C} Aff(y_i, c, c(i)) = \sum_{i=1}^{N}\sum_{c=1}^{C} \frac{(||y_i - \mu_c||^2)^\beta - (||y_i - \mu_{c(i)}||^2)^\beta}{(||y_i - \mu_c||^2)^\beta + (||y_i - \mu_{c(i)}||^2)^\beta}$$

qui est un critère que l'on cherchera à maximiser

20.2 écriture du gradient

Pour permettre l'optimisation par descente de gradient il faut exprimer la dérivée du critère par rapport aux poids. Les y_i sont les transformations des x_i par un kernel gaussien,
ainsi $||y_i - y_j||^2 = 2 - 2e^{-||x'_i - x'_j||}$ où $(x'_i)_d = (x_i)_d w_d$. On trouve donc :

$$K(i,j) = e^{-\sum_{d=1}^{D}(x_i(d)-x_j(d))^2 w_d^2}$$

$$\frac{\partial K(i,j)}{\partial w_d} = -2w_d(x_i(d) - x_j(d))^2 K(i,j)$$

$$||y_i - \mu_c||^2 = ||y_i||^2 + ||\mu_c||^2 - 2y_i.\mu_c = 1 + \frac{1}{N_c^2}\sum_{\substack{n=1\\L(n)=c}}^{N}\sum_{\substack{m=1\\L(m)=c}}^{N} K(n,m) - \frac{2}{N_c}\sum_{\substack{n=1\\L(n)=c}}^{N} K(n,i)$$

$$\frac{\partial||y_i - \mu_c||^2}{\partial w_d} = \frac{1}{N_c^2}\sum_{\substack{n=1\\L(n)=c}}^{N}\sum_{\substack{m=1\\L(m)=c}}^{N} \frac{\partial K(n,m)}{\partial w_d} - \frac{2}{N_c}\sum_{\substack{n=1\\L(n)=c}}^{N} \frac{\partial K(n,i)}{\partial w_d}$$

il est a noté qu'en choisissant $K(i,j) = x_i x_j^T$, c'est à dire en utilisant le « kernel identité » on trouve un algorithme de descente de gradient stochastique pour l'optimisation du Fisher criterion, à comparer avec l'algorithme donné généralement qui permet de trouver la solution explicitement mais uniquement sur un ensemble de donné connu dans son ensemble, donc inapplicable ici.

20.3 minimum locaux

Nous avons pu constater numérique que dans le cas général, des minimum locaux existent pour chacun de ces critères. Néanmoins, nous proposons d'utiliser une descente de gradient pour tenter de minimiser ces critères.

20.4 optimisation pour Kernel K-means ou pour Normalized cut

Les critères donnés précédemment sont des critères de bonne répartition des points autour de leur centroïde. Ils peuvent donc être calculés dans n'importe quel espace où les données sont supposées suivre ce genre de répartition. Notamment, lorsque l'on fait un clustering avec l'algorithme K-means, kernel K-means ou Normalized Cut, on fait implicitement cette supposition sur les données. Il est donc cohérent de tenter d'optimiser la bonne répartition des points autour de leur centroïde respectivement dans l'espace original, dans l'espace kernel (critère Fisher ou quotient), et dans l'espace du Laplacien (critères weighted Fisher et weighted quotient) pour ensuite espérer obtenir des gains de performances en utilisant respectivement comme algorithme de clustering : K-means, kernel K-means, et Normalized Cut.

On utilise la dénomination weighted-Fisher et weighted-quotient puisqu'il s'agit bien du Fisher criterion et du quotient criterion défini plus haut, mais dans l'espace engendré par le laplacien de la matrice d'adjacence, i.e celui où fait place l'algorithme de spectral clustering utilisé pour minimiser le critère normalized Cut, équivalent au critère weighted (kernel) K-means.

Atention : dans cet espace les centroïdes sont situés au barycentre plutôt qu'au centre de leur classe, puisque les points sont affectés d'une pondération qui peut varier de manière importante.

La proposition que nous faisons ici, que nous n'avons malheureusement pas réussi à démontrer entièrement expérimentalement, est qu'il existe des critères permettant d'optimiser la répartition des points dans l'espace du Laplacien et d'obtenir des gains de performances avec les algorithmes utilisant le critère Normalized Cut.

Pour plus d'informations sur les approches existentes en matière d'apprentissage de kernel, nous faisons référence à un article de Xiong [13, Xiong2005] utilisant le Fisher criterion pour optimiser les paramètres d'une fonction kernel.

20.5 optimisation du critère pour Normalized Cut

note : Il s'agit d'optimiser la répartition des points dans l'espace induit par le Laplacien de la matrice de similarité

La base de donnée utilisée est celle ne comprenant que l'octave 4 présentée au paragraphe 12.

Les features que l'on cherche à pondérér sont, dans l'ordre vu sur le graphique :

1. EnvComp : 128 points en fréquence, enveloppe estimée par interpolation linéaire des partiels, puis compressée avec une compression exponentielle de paramètre 0.4.
   Les points d'échantillonnage sont répartis linéairement de 50 à 8000Hz.
2. EnvLog : même chose avec une compresion logarithmique.
3. AmpComp : amplitude compressée (compression exponentielle 0.4) des 30 premiers partiels
4. AmpCompLP3 : amplitude compressée (compression exponentielle 0.4) des 30 premiers partiels convoluées par un filtre de réponse impulsionnelle $[0.5, 1, 0.5]$.
5. AmpCompLP5 : amplitude compressée (compression exponentielle 0.4) des 30 premiers partiels convoluées par un filtre de réponse impulsionnelle $[0.25, 0.75, 1, 0.75, 0.25]$.
6. logarithme en base 2 de la fréquence fondamentale

Soit au total $128 \times 2 + 30 \times 3 + 1 = 347$ features.

Voici un premier affichage (figure 27) de notre algorithme de descente de gradient cherchant à minimiser le weighted-fisher criterion (courbe rouge) et à maximiser le weighted-quotient criterion (courbe bleue). Les résultats sont ici très bons, mais nous montrons à la fin de cette partie un exemple où les résultats sont beaucoup plus mitigés, voir mauvais. On peut voir que les valeurs de weighted-fisher criterion et de weighted-quotient criterion rapportées

à celles de la pondération initiale deviennent progressivement au cours des itérations inférieures et supérieures à 1. Cela montre donc qu'il est possible, sur une base de donnée difficile, d'optimiser chacun de ces critères. On remarque également que la pondération qui optimise le weighted Fisher criterion maximise aussi le weighted quotient criterion, et vice-versa, montrant un lien certain entre ces deux mesures de bonne répartition.

En haut nous pouvons voir les pondérations obtenues après 400 itérations. Celles-ci sont plutôt proches l'une de l'autre, montrant un lien certain entre les 2 critères. Pour rendre la comparaison plus facile, on voit en vert l'initialisation de la descente de gradient (et donc tout le chemin parcouru en 400 itérations). Il s'agit d'une descente de gradient stochastique puisqu'il est inenvisageable de traiter toute la base de donnée à chaque itération, celle-ci étant virtuellement très grande.

Sur cette figure on voit que le Fisher Criterion semble donner de meilleurs résultats que le critère quotient. On vérifie sur des scores de clustering que les Det Curves sont d'assez bons indicateurs de qualité d'une mesure de similarité, et que les pondérations optimisées donnent de meilleurs résultats :

Néanmoins, on peut voir que le critère donnant les meilleurs résultats en terme de clustering est le quotient criterion. En effet nous avons développé ce critère en minimisant avec la fonction *tanh* le poids donné aux points très proches ou très loins du centroïde de leur classe.

Nous avons donc montré qu'optimiser le fisher criterion ou le quotient criterion pour trouver une meilleure pondération des features avant l'application d'un kernel gaussien fait sens et peut améliorer les résultats de clustering.

Sur la figure 30 on peut voir que l'optimisation avec d'autres représentations (les MFCC et les MFC-log ont été ajoutés) et avec une autre initialisation ne donnait pas du tout des résultats francs comme espérés, mauvais résultats confirmés en clustering (où le quotient criterion donne tout de même honoraleblement exactement le même score que la pondération initiale).

Cela relance donc la question : existe-t-il un critère dans l'espace du Laplacien qu'il est pertinent d'optimiser dans le cas général ?

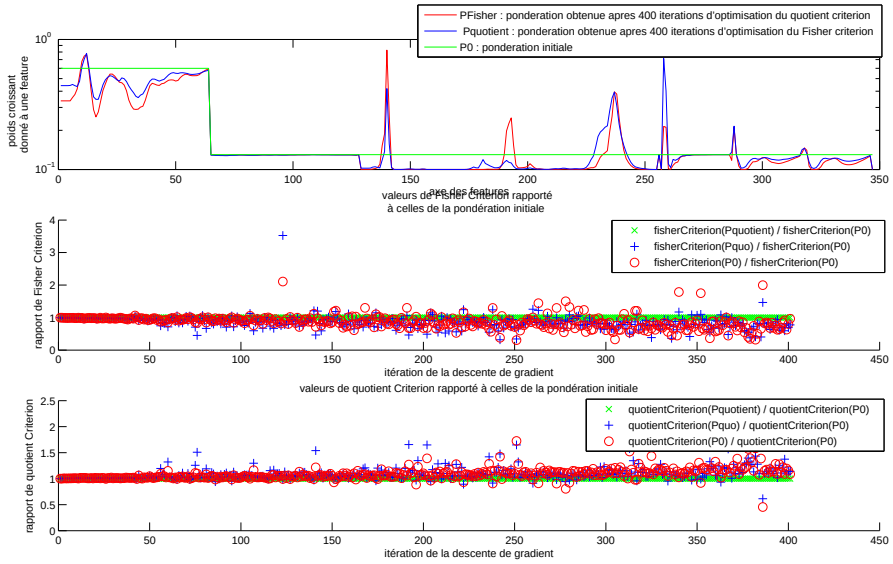

FIGURE 27 – Affichage des pondératios obtenues après 400 itérations de descente de gradient . Critères utiliés : weighted-Fisher criterion et weighted-quotient criterion.

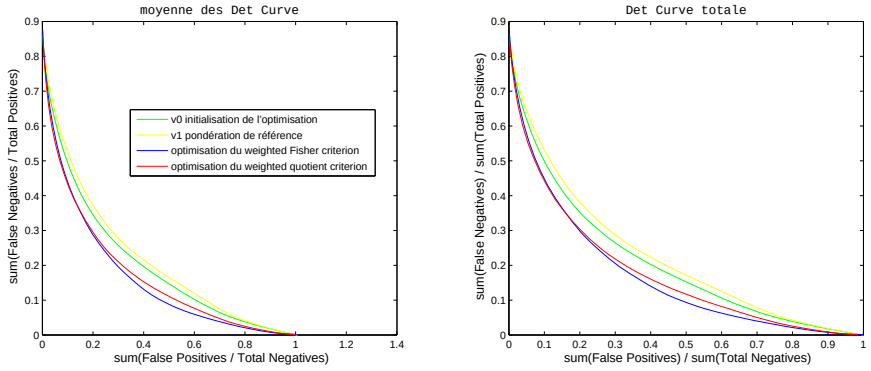

FIGURE 28 – Det Curves moyennes obtenues sur la base de donnée comprenant tous les mélanges d'instruments, d'interprètes et de sous-ensembles de notes de l'octave 4

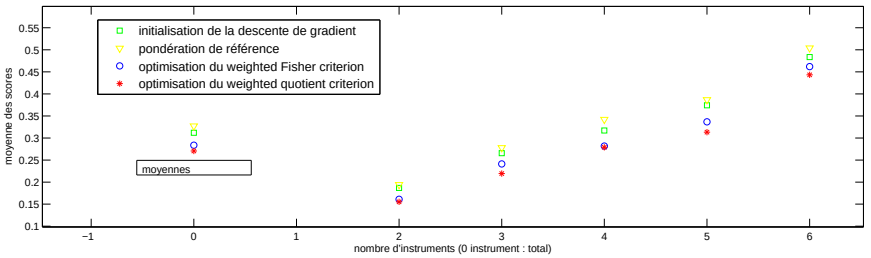

FIGURE 29 – Scores de clustering moyens obtenues sur la base de donnée comprenant tous les mélanges d'instruments, d'interprètes et de sous-ensembles de notes de l'octave 4 (l'algorithme utilisé est Normalized Cut - Spectral Clustering).

20.6 optimisation pour kernel K-means

Sur la figure 31 le Fisher criterion semble donner de très mauvais résultats, alors que le quotient criterion semble plutôt bon même si la Det Curve obtenue croise celle de l'initialisation.

On vérifie figure 32 que le quotient criterion permet bien d'optimiser la bonne répartition des points dans l'espace kernel, et tend à améliorer les performances en clustering si l'algorithme kernel K-means est utilisé.

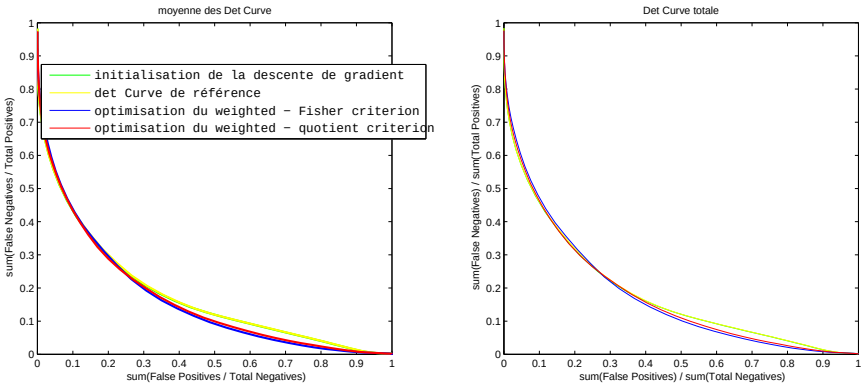

FIGURE 30 – Det Curves obtenues avec l'algorithme normalized Cut (enveloppe compression exp, log, amplitudes partiels compression exp non moyennés, moyennés sur 5 et 9 points, MFCC, MFC-log, et log fréquence fondamentale

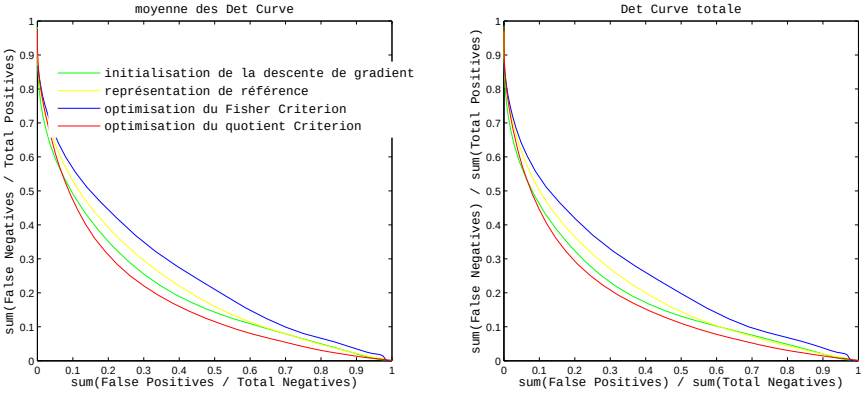

FIGURE 31 – Det Curves obtenues avec une autre représentation (enveloppe compression exp, log, amplitudes partiels compression exp non moyennés, moyennés sur 5 et 9 points, MFCC, MFC-log, et log fréquence fondamentale

En utilisant par exemple l'algorithme normalized Cut qui travaille dans l'espace engendré par le laplacien de la matrice de Gram, le gain de performance n'est plus notable, ce qui peut sembler logique puisque le critère optimisé est un critère lié directement à l'espace kernel et non par à l'espace du laplacien de la matrice de Gram.

Cependant, comme tout algorithme d'apprentissage, nous avons élaboré une base de test avec les échantillons provenant de la base de donnée « IOWA », et les performances obtenues avec la mesure de similarité pondérée sont moins bonnes qu'avant optimisation, aussi bien pour le Fisher criterion que pour le critère quotient. Le problème du sur-apprentissage se pose donc, et des études complémentaires sont nécessaires pour prouver l'intéret d'une telle pondération dans un cadre de classification ou de clustering.

21 Estimation des paramètres du modèle de timbre en polyphonique

Le problème est le suivant : à partir du signal et des F0s qui sont supposées correctes, comment estimer les enveloppes fréquentielles et leur évolution temporelle.

Un problème plus général serait d'estimer la distribution énergétique dans le plan temps fréquence des diverses sources/F0.

Si les sources sont supposées approximativement harmoniques, ce problème se réduit alors à la « résolution des partiels superposés » et ne peut être fait qu'au niveau global.

21.1 méthode fréquentielle pour résoudre l'overlap

Une simplification souvent faite du problème consiste à ne pas faire de supposition sur l'évolution temporelle des enveloppes, i.e de les considérer indépendantes, et de travailler alors sur chaque trame indépendament.

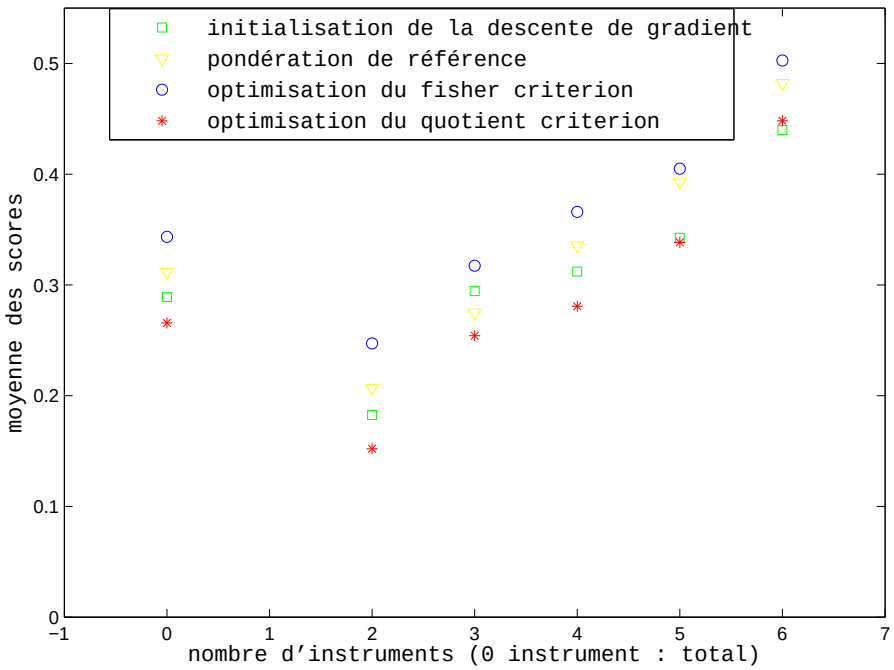

FIGURE 32 – Performances de clustering obtenues avec l'algorithme kernel K-means (enveloppe compression exp, log, amplitudes partiels compression exp non moyennés, moyennés sur 5 et 9 points, MFCC, MFC-log, et log fréquence fondamentale

La supposition qui est faite souvent (par exemple chez [4, Yeh2010]) est que les partiels de chaque source suivent une courbe régulière en fréquence. Il s'agit ni plus ni moins du modèle source filtre,

sans la contrainte temporelle que le filtre doit varier lentement. Cette contrainte est d'ailleurs introduite chez [1, Heittola2009] on encore chez citeMiyamoto2008 directement dans la détection de F0.

Sans contrainte temporelle, la solution trouvée peut être mauvaise, et engendrer des distrutions énergétiques chaotiques temporellement, ou encore « bruitées ». Introduire la contrainte temporelle directement dans la détection de F0 est certainement une bonne chose, mais est aussi certainement trop ambitieux, à savoir qu'il risque d'être difficile de trouver un algorithme qui résolve tous les problèmes de l'analyse multipitch en même temps.

21.2 méthode temporelle proposée

La méthode proposée ici consiste à introduire des variables ressort et des contraintes temporelles, pour le moment en fixant empiriquement les paramètres et en utilisant des distributions gaussiennes, mais dans le futur il serait bon de formuler de manière probabiliste les distributions mises en jeu et définir un likelihood global pour les variables ressorts, que l'on tentera d'optimiser.

Après analyse sinusoïdale, si un pic d'amplitude d'amplitude A est supposé cacher plusieurs pics superposés, alors on crée autant de variables $x_1 \cdots x_n$ que de sources supposées se superposer, et on introduit des contraintes :

$$A \approx \sum_n x_n$$

Par ailleurs, si x_k et x_n représentent l'amplitude du même partiel d'une même source à deux trames successives, on ajoute la contrainte :

$$x_k \approx x_n$$

Ces contraintes formulées de manière probabiliste devraient permettre de définir un likelihood bien fondé, que l'on tentera d'optimiser de manière globale.

L'évaluation de cet algorithme pourra se faire simplement mélangeant plusieurs sources monophoniques, puis en calculant un Signal to Noise Ratio.

– Modèle 1.a

$$x(p) = \mu_c(p) + \epsilon$$

$$y \in c \text{ si } ||x - \mu_c|| = \min_k ||x - \mu_k||$$

– Modèle 1.b

$$x(p) = \mu_c(p) + \epsilon$$

$$y \in c \text{ si } ||x - \mu_c|| = \min_k ||x - \mu_k||$$

Cinquième partie

Conclusion et Bibliographie

Dans ce mémoire nous avons étudié et chiffré plusieurs méthodes concernant le problème de l'élaboration d'une mesure de similarité timbrale, la formation/clustering de flux ou de sources, nous avons également tenté d'appliquer les méthodes à noyaux pour former des cluster cohérents avec la mesure de similarité proposée.

Nous avons proposé plusieurs algorithmes nouveaux : l'un concernant la pondération et la sélection de feature dans un cadre de clustering, un autre pour l'optimisation visuelle de mesure de similarité à l'aide de Det Curves, et enfin plusieurs algorithmes de type K-means et GMM-EM modifiés pour l'estimation d'enveloppe, le clustering de vecteurs d'amplitudes de partiels et la classification d'instrument.

Il s'agit d'un travail non abouti en ce sens que nous n'avons pas chiffré les performances et la robustesse des algorithmes présentés dans un cadre d'analyse polyphonique, et qu'en classification nous encore très loin des algorithmes de l'état de l'art, qui font usage de nombre de descripteurs dont beaucoup ne sont pas robustes en cas de mélange polyphonique.

Plusieurs aspects théoriques et pratiques auxquels nous avons en parti été confrontés n'ont pas été étudiés, beaucoup paraissent a priori difficiles. Dans le désordre nous pouvons citer :

1. utiliser les méthodes à noyau et la pondération de descripteurs pour tenter d'atteindre les performances de l'état de l'art en classification,
2. chercher à optimiser le Fisher criterion dans l'espace original en utilisant le kernel identité,
3. établir un modèle de fonction de compression, et mettre en oeuvre un algorithme fonctionnel pour l'optimisation de la fonction de compression dans le cadre de clustering timbral et évaluer les applications possibles en classification,
4. montrer rigoureusement l'utilité du modèle source-filtre multiplicatif ou additif en classification ou en clustering
5. étudier les applications possibles des algorithmes déformés ou avec modèle de source au traitement de la parole et à l'estimation dynamique des modèles de voyelles,
6. l'évaluation des méthodes de l'état de l'art d'apprentissage de kernel dans le cadre du clustering d'instruments,
7. tenter d'unifier les méthodes de qualification de mesures de similarités utilisées dans ce rapport : Det curves, clustering, critères Fisher et quotient,
8. montrer la pertinence de l'optimisation de critères dans l'espace du laplacien d'une matrice de similarité,
9. montrer que pour certaines distributions des données certain critères peuvent être plus robustes que d'autres comme le Fisher criterion,
10. évaluer en proposant des critères d'évaluation les méthodes de résolution de partiels superposés.

Références

[1] T. Heittola A. Klapuri T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. 2009.

[2] J.J. Burred A. Roebel T. Sikora. Polyphonic musical instrument recognition based on a dynamic model of the spectral envelope. 2009.

[3] Antti Eronen Anssi Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. 2000.

[4] Chunghsin Yeh Axel Roebel and Xavier Rodet. Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals. 2010.

[5] Juan Jose Burred Axel Roebel Thomas Sikora. Dynamic spectral envelope modeling for the analysis of musical instrument sounds. 2009.

[6] Graham Grindlay Daniel P. W. Ellis. Multi-voice polyphonic music transcription using eigeninstruments. 2009.

[7] Peter D. Wentzell Daren T. Andrews David C. Hamilton Klaas Faber Bruce R. Kowalski. Maximum likelihood principal component analysis. 1997.

[8] Daniel D. Lee H. Sebastian Seung. Algorithms for non-negative matrix factorization. 2001.

[9] Kenichi Miyamoto Hirokazu Kameokay Takuya Nishimoto Nobutaka Ono Shigeki Sagayama. Harmonic-temporal-timbral clustering (httc) for the analysis of multi-instrument polyphonic music signals. 2008.

[10] Masataka Goto Hiroki Hashiguchi Takuichi Nishimura Ryuichi Oka. Rwc music database : Popular, classical, and jazz music databases. 2003.

[11] Stella X. Yu Jianbo Shi. Multiclass spectral clustering. 2003.

[12] A.Y. Ng M. Jordan Y. Weiss. On spectral clustering : Analysis and an algorithm. 2001.

[13] Huilin Xiong M. N. S. Swamy M. Omair Ahmad. Optimizing the kernel in the empirical feature space. 2005.

[14] Tetsuro Kitahara Masataka Goto Hiroshi G. Okunoy. Musical instrument identification based on f0-dependent multivariate normal distribution. 2003.

[15] Lu Ye Minas E. Spetsakis. Clustering on unobserved data using mixture of gaussians. 2003.

[16] D. Blondel Ngoc-Diep Ho Paul van Dooren. Weighted nonnegative matrix factorization and face feature extraction. 2007.

[17] Tomoyoshi Kinoshita Shuichi Sakai Hidehiko Tanaka. Musical sound source identification based on frequency component adaptation. 2006.

[18] N Srebro T Jaakkola. Weighted low rank approximations. 2003.

[19] Chris Ding Tao Li Michael I. Jordan. Convex and semi-nonnegative matrix factorizations. 2006.

[20] Chris Ding Xiaofeng He. K -means clustering via principal component analysis. 2004.

[21] Inderjit S. Dhillon Yuqiang Guan and Brian Kulis. Weighted graph cuts without eigenvectors : A multilevel approach. 2007.

[22] Sean Borman. The expectation maximization algorithm, a short tutorial. 2004.

[23] Anssi Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. 2008.

[24] Bernhard Scholkopf. Input space versus feature space in kernel-based methods. 1999.

Sixième partie

Annexe

Dans cette annexe se trouvent la plupart des démonstrations et formules manquantes dans les sections précédentes, et le texte les accompagnant a été réduit à son minimum.

22 K-means

K-means est de loin, avec toutes ses variantes, l'algorithme de clustering le plus utilisé. Pourtant, dans le cas général, le problème de décision de « savoir si oui ou non une partition est optimale au sens des K-means » est NP-complet, et donc difficile à résoudre.

22.1 dans l'espace de description

Soit un ensemble de points $x_{i=1\cdots N}$ de $\mathbb{R}^D$ et C le nombre de classes supposé. Le problème K-means consiste à trouver une partition, $S_1 \cdots S_C$ de l'ensemble des observations minimisant le critère :

$$J = \sum_{c=1}^{C} \sum_{i \in S_c} ||x_i - \mu_c||^2$$

μ_c est alors le centroïde de la classe c, et connaissant la partition on a :

$$\mu_c = \frac{1}{|S_c|} \sum_{i \in S_c} x_i$$

L'algorithme procède en répétant les 2 étapes :
- améliorer le partionnement connaissant les centroïdes
- recalculer les centroïdes connaissant le partionnement

c'est donc un algorithme de recherche locale, où la relation de proximité entre les solutions acceptables est défini comme : deux solutions acceptables sont proches si elles ont en commun la partition ou les centroïdes

dans l'expression de J en développant $||x_i - \mu_c||^2 = ||x_i||^2 + ||\mu_c||^2 - 2x_i.\mu_c$ on trouve

$$J = \sum_{c=1}^{C} \sum_{i \in S_c} \left(||x_i||^2 + \frac{1}{N^2} \sum_{j,k \in S_c} x_j.x_k - \frac{2}{|S_c|} \sum_{j \in S_c} x_i.x_j \right) = \sum_{c=1}^{C} \sum_{i \in S_c} ||x_i||^2 + \sum_{c=1}^{C} \frac{1}{|S_c|} \sum_{j,k \in S_c} x_j.x_k - \sum_{c=1}^{C} \frac{2}{|S_c|} \sum_{i,j \in S_c} x_i.x_j$$

$\sum_{c=1}^{C} \sum_{i \in S_c} ||x_i||^2$ étant un terme constant, on peut remplacer J par

$$J' = \sum_{c=1}^{C} \frac{1}{|S_c|} \sum_{i,j \in S_c} x_i.x_j$$

qui devient le critère à maximiser.

si une observation revient plusieurs fois on peut utilement la spécifier une seule fois et indiquer sa multiplicité par un coefficient, ou poids. Cela peut se généraliser à des poids rationnels, voir réels.
on obtient alors (criètre des weighted K-means) :

$$J = \sum_{c=1}^{C} \sum_{i \in S_c} w_i ||x_i - \mu_c||^2$$

conssaissant la partition, le centroïde devient :

$$\mu_c = \frac{\sum_{i \in S_c} w_i x_i}{\sum_{i \in S_c} w_i}$$

22.2 Kernel K-means

Si une fonction de redescription envoie un point x_i de $\mathbb{R}^D$ dans un espace euclidien donc muni d'un produit scalaire et de la distance euclidienne, on a alors :

$$J = \sum_{c=1}^{C} \sum_{i \in S_c} ||\phi(x_i) - \mu_c||^2$$

$$= \sum_{c=1}^{C} \sum_{i \in S_c} \phi(x_i).\phi(x_i) + \mu_c.\mu_c - 2\phi(x_i).\mu_c$$

on définit la fonction noyau comme étant :

$$K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$$

$$K(x_i, x_j) = \phi(x_i).\phi(x_j)$$

enfin μ_c étant le centroïde de la classe c dans l'espace de redescription qui est un espace euclidien on a

$$\mu_c = \frac{1}{|S_c|} \sum_{i \in S_c} \phi(x_i)$$

si bien qu'en remplaçant dans l'équation précédante on peut écrire :

$$J = \sum_{c=1}^{C} \sum_{i \in S_c} \left(K(x_i, x_i) + \frac{1}{|S_c|^2} \sum_{j \in S_c} \sum_{k \in S_c} K(x_j, x_k) - \frac{2}{|S_c|} \sum_{j \in S_c} K(x_i, x_j) \right)$$

minimiser J est alors équivalent à maximiser :

$$J' = \sum_{c=1}^{C} \frac{1}{|S_c|} \sum_{i,j \in S_c} K(x_i, x_j)$$

23 Algorithmes sur les graphes

23.1 Normalized Cut

soit un graphe non orienté $(\mathcal{V}, E, A)$ où les sommets sont des points de l'espace et où le poids d'une arrête $a(u,t)$ représente l'affinité entre les points u et t.

23.1.1 cas à 2 classes

soit une partition $\mathcal{A}, \mathcal{B}$ de $\mathcal{V}$

$$cut(A,B) = \sum_{u \in \mathcal{A}, t \in \mathcal{B}} a(u,t)$$

$$assoc(A) = \sum_{u \in \mathcal{A}, t \in \mathcal{V}} a(u,t)$$

$$Ncut(A,B) = \frac{cut(\mathcal{A},\mathcal{B})}{assoc(\mathcal{A})} + \frac{cut(\mathcal{A},\mathcal{B})}{assoc(\mathcal{B})}$$

résoudre le problème normalized-cut consiste à trouver la partition qui minimise $Ncut(\mathcal{A},\mathcal{B})$

23.1.2 cas à K classes

en écrivant que

$$Intra(\mathcal{A}) = \sum_{(u,t)\in\mathcal{A}^2} a(u,t)$$

on trouve que

$$Ncut(\mathcal{A}_1, \cdots \mathcal{A}_K) = \sum_{k=1}^{K} \frac{cut(\mathcal{A}_k, \mathcal{V}/\mathcal{A}_k)}{assoc(\mathcal{A}_k)} = \sum_{k=1}^{K} \frac{assoc(\mathcal{A}_k) - Intra(\mathcal{A}_k, \mathcal{A}_k)}{assoc(\mathcal{A}_k)}$$

$$\min Ncut(\mathcal{A}_1, \cdots \mathcal{A}_K) \Leftrightarrow \max \sum_{k=1}^{K} \frac{\text{Intra}(\mathcal{A}_k)}{assoc(\mathcal{A}_k)}$$

on peut bien sûr parcourir l'abre des partitions possibles avec un algorithme de type branch and cut, mais le temps de calcul à partir d'un certain nombre de sommet peut devenir prohibitif.

Le plus souvent, pour optimiser le critère Ncut (ou d'autres critères tel que «ratio cut» ou «ratio association») on utilise l'algorithme dit de «spectral clustering».

avec x_k le vecteur colonne indicateur de la partition k ($x_k(j) = 1$ si v_j appartient à la partition k), et D la matrice diagonale qui contient les degrés de chaque sommet (somme des poids des arêtes qui partent d'un sommet) on obtient les relations suivantes :

$$Intra(\mathcal{A}) = x_k^T A x_k$$
$$Assoc(\mathcal{A}) = x_k^T D x_k$$

$$\sum_{k=1}^{K} \frac{\text{Intra}(\mathcal{A}_k)}{assoc(\mathcal{A}_k)} = \sum_{k=1}^{K} \frac{x_k^T A x_k}{x_k^T D x_k} = \text{tr}(X^T AX(X^T DX)^{-1})$$

avec $X = [x_1 \cdots x_K]$. La dernière égalité est intuitive une fois que l'on a remarqué que $X^T DX$ est une matrice diagonale, X étant une matrice indicatrice de partition.
En utilisant $\text{tr}(BA) = \text{tr}(AB)$ on trouve :

$$\text{tr}(X^T AX(X^T DX)^{-1}) = \text{tr}((X^T DX)^{-1/2} X^T AX(X^T DX)^{-1/2}) = \text{tr}(Y^T D^{-1/2} AD^{-1/2} Y)$$

avec $Y = D^{1/2} X(X^T DX)^{-1/2}$ ou encore plus explicitement :

$$Y(n,k) = \begin{cases} \sqrt{D(n)}/\sqrt{\sum_{n\in k} D(n)} & \text{si } n \in k \\ 0 & \text{sinon} \end{cases}$$

les colonnes de Y sont orthogonales :

$$Y^T Y = (X^T DX)^{-1/2} X^T D^{1/2} D^{1/2} X(X^T DX)^{-1/2} = Id_k$$

De plus, pour que Y soit une solution potentielle à notre problème, il faut que :

$$Y \geq 0$$
$$Y 1_k = \text{diag}(D)$$

La condition de positivité ajoutée à l'orthogonalité nous assure qu'une seule valeur par ligne est non nulle.
Si l'on enlève l'une ou plusieurs de ces contraintes on obtient une matrice qui peut ne plus être en rapport avec notre problème, ou même ne plus représenter une partition.

23.1.3 Spectral Clustering

Le critère à maximiser est donc :

$$J = \mathrm{tr}(Y^T D^{-1/2} A D^{-1/2} Y)$$

Si on ne garde que la contrainte d'orthogonalité ($Y^T Y = Id$), on obtient une pseudo-affectation plutôt qu'une partition, avec des valeurs d'affectation qui peuvent être nulles ou même négatives.

La matrice qui maximise J est alors composée des K premiers vecteurs propres de $D^{-1/2} A D^{-1/2}$ (en classant les valeurs propres, qui sont positives, par ordre décroissant). Ces vecteurs propres sont ensuite transformés en une indicatrice de partition.

La méthode la plus souvent utilisée constiste à chercher une matrice indicatrice, aussi proche que possible de la matrice des vecteurs propres (voir [12, Weiss2001] ou encore [11, Shi2003]).

Cet algorithme fournit donc un moyen simple, rapide si la matrice à diagonaliser n'est pas trop grande, et souvent efficace même si sous-optimal, de minimiser le critère Normalized Cut. Il peut également être adapté à d'autres critères, tel que ratioCut ou ratioAssociation [21, Kulis2007].

23.1.4 équivalence entre Normalized Cut et weighted Kernel K-means

Cette équivalence est présentée dans plusieurs articles, donc [21, Kulis2007]. On écrit le critère des weighted kernel K-means :

$$J = \sum_{c=1}^{C} \sum_{i \in S_c} w_i ||\phi(x_i) - \mu_c||^2$$

$$= \sum_{c=1}^{C} \sum_{i \in S_c} w_i ||\phi(x_i)||^2 + \sum_{c=1}^{C} \sum_{i \in S_c} w_i \sum_{j,k \in S_c} \frac{w_j w_k \phi(x_j).\phi(x_k)}{(\sum_{l \in S_c} w_l)^2} - 2 \sum_{c=1}^{C} \sum_{i \in S_c} w_i \sum_{j \in S_c} \frac{w_j \phi(x_i).\phi(x_j)}{\sum_{l \in S_c} w_l}$$

$$= \sum_{c=1}^{C} \sum_{i \in S_c} w_i ||\phi(x_i)||^2 - \sum_{c=1}^{C} \sum_{i \in S_c} w_i \sum_{j \in S_c} \frac{w_j \phi(x_i).\phi(x_j)}{\sum_{l \in S_c} w_l}$$

on retrouve donc un terme constant et :

$$J' = \sum_{c=1}^{C} \frac{\sum_{i,j \in S_c} w_i w_j \phi(x_i).\phi(x_j)}{\sum_{j \in S_c} w_j}$$

devient le critère à maximiser

en introduisant les x_c vecteurs colonnes indicateur de la partition, $X = [x_1 \cdots X_C]$,

W la matrice diagonale des poids et K la matrice de gram ($K(i,j) = \phi(x_i).\phi(x_j)$) on a :

$$J' = \sum_{c=1}^{C} \frac{x_c^T W K W x_c}{x_c^T W x_c} = tr(X^T W K W X (X^T W X)^{-1})$$

on remarque qu'en posant $A = WKW$ et $D = W$ on a que J' est équivalent au critère J trouvé pour Normliazed Cut. Cependant weighted kernel K-means est plus général que Normalized Cut dans le sens où il n'y a pas la contrainte que W soit la matrice des sommes des lignes de WKW. Ncut et weighted kernel K-means ne sont donc pas équivalent mais on a plutôt : « Ncut inclus dans la classe des problèmes weighted kernel K-means ».

Le critère normalized Cut repose donc sur la supposition que les points (envoyés dans l'espace Kernel en utilisant comme matrice de gram $K = D^{-1} A D^{-1}$) sont regroupés en des boules formant des classes.

Dans notre cas, la matrice A est calculée à partir des données brutes, à l'aide d'un kernel modélisant une mesure de similarité. Le choix du kernel utilisé est donc primodial, et selon le cas cette supposition sur la géométrie des classes sera pertinente ou non.

pour transformer un problème Ncut connaissant la matrice d'adjacence A en un problème weighted kernel k-means de poids W et de matrice de gram K on pose

$$K = W^{-1} A W^{-1}$$

avec $w_k = \sum_i A(i,k)$

le critère des weighted kernel k-means à minimiser est alors :

$$J = \sum_c \sum_{k \in c} w_k ||\phi(x_k) - \mu_c||^2$$

avec $K(i,j) = \phi(x_i).\phi(x_j)$

$$||\phi(x_k) - \mu_c||^2 = ||\phi(x_k)||^2 + ||\mu_c||^2 - 2\phi(x_k)\mu_c$$

$$\mu_c = \frac{\sum_{k \in c} w_k \phi(x_k)}{\sum_{k \in c} w_k}$$

$$||\phi(x_k) - \mu_c||^2 = K(k,k) + \frac{\sum_{i,j \in c} w_i w_j K(i,j)}{(\sum_{i \in c} w_i)^2} - 2\frac{\sum_{i \in c} w_i K(i,k)}{\sum_{i \in c} w_i}$$

$$= \frac{A(k,k)}{{w_k}^2} + \frac{\sum_{i,j \in c} A(i,j)}{(\sum_{i \in c} w_i)^2} - \frac{2}{w_k}\frac{\sum_{i \in c} A(i,k)}{\sum_{i \in c} w_i}$$

23.2 un algorithme multi-niveau sans calcul de vecteurs propres

On peut réexprimer Normalized Cut comme un problème weighted (kernel) K-means. Le problème est que l'algorithme itératif souvent utilisé pour K-means soufre dans ce cas de l'existence de beaucoup de minimum locaux. Pour palier à ce problème, [21, Kulis2007] propose de simplifier le graphe à l'extrême en regroupant des sommets, de réaliser un premier clustering «grossier», plus d'affiner celui-ci en même temps que les super-sommets sont divisés.

A la première étape, deux sommets sont fusionnés si l'arête qui les lie a un poids suffisamment fort. A la seconde étape, un super-sommet est divisé en plusieurs sommets (qui peuvent être également des groupes de sommets), et une recherche locale (mélangeant une itération de type K-means batch et une sélection du meilleur sommet à déplacer) est utilisée pour faire diminuer le critère weighted (kernel) K-means.

Cet algorithme présente l'avantage de ne pas calculer de vecteurs propres, avantage non négligeabl quand le nombre de sommets devient très grand.

23.3 Conclusion

d'un côté on peut utiliser un algorithme de recherche locale itératif pour optimiser le critère, qui présente l'avantage d'être facile à implémenter, de ne pas calculer de vecteurs propres, et d'être très efficace dans le cas de données effectivement réparties en des classes bien distinctes.

dans le cas de données plus compliquées (ce qui est souvent le cas), il est préférable d'utiliser l'algorithme de spectral clustering qui soufre moins des minimum, mais qui est difficilement implémentable si les données sont trop nombreuses.

24 Algorithme K-means déformé

on généralise un peu le critère weighted K-means en ajoutant une distance de Mahanalobis propre à chaque point :

$$J = \sum_c \sum_{i \in c} (\mu_c - x_i)^T M_i (\mu_c - x_i)$$

On exige que les M_i soient toutes inversibles, symmétriques, et définies positives, ce qui implique que les matrices $\mathcal{M}_c = \sum_{i \in c} M_i$ le sont également, pour n'importe quelle partition.

Connaissant la partition, les centroïdes s'expriment comme :

$$\mu_c = \mathcal{M}_c^{-1} \bar{x}$$

avec

$$\mathcal{M}_c = \sum_{i \in c} M_i$$

$$\bar{x} = \sum_{i \in c} M_i x_i$$

On peut appliquer l'algorithme habituel des K-means, qui convergera en un nombre fini d'itération.

En effet, on cherche :

$$\mu_c = \underset{\mu}{\operatorname{argmin}}\ E$$

$$E = \sum_{i \in c} (\mu - x_i)^T M_i (\mu - x_i)$$

On écrit que le gradient doit s'annuler :

$$\nabla E = \sum_{i \in c} \left(M_i(\mu - x_i) + M_i^T(\mu - x_i) \right) = 0$$

Comme $M_i = M_i^T$, on obtient :

$$\left(\sum_{i \in c} M_i \right) \mu = \sum_{i \in c} M_i x_i$$

$$\mathcal{M}_c \mu = \sum_{i \in c} M_i x_i$$

$$\mu = \mathcal{M}_c^{-1} \bar{x}$$

Le gradient ne s'annule qu'en un point car E est une forme quadratique, convexe en les composantes de μ et admet un minimum.

En utilisant l'algorithme itératif habituel des K-means, le critère diminue strictement à chaque itération : à l'étape du calcul de la partition, un point qui n'est pas associé à son centroïde le plus proche sera réaffecté, donc fera diminuer le critère, si tous les points sont déjà affectés aux centroïdes les plus proches, l'algorithme s'arrête. A l'étape du calcul des centroïdes, l'équation ci-dessus montre que le critère ne peut augmenter.

On parcourt donc l'ensemble des partitions, qui est fini, en respectant le préordre strict défini par le critère J, ce qui montre la convergence de l'algorithme K-means déformé.

25 PCA déformé

Le lemme donné au paragraphe 29.2.1 qui découle directement de la démonstration d'optimalité de la transformation PCA :

soit $X : I \times D$ $W : I \times K$ $H : K \times D$ $P : D \times K$ $M : I \times D$

$$\text{si} \quad (W, H) = \underset{(W,H)}{\operatorname{argmin}} ||X - WH||^2$$

$$\text{et} \quad P = \underset{P}{\operatorname{argmin}} ||WH - WHPP^T||^2$$

$$\text{alors} \quad P = \underset{P}{\operatorname{argmin}} ||X - XPP^T||^2$$

N'est plus applicable lorsque l'on ajoute une pondération M_{id}. Cela signifie que

$$\min J_1 = \sum_i \sum_d (X_{id} - \sum_k W_{ik} H_{kd})^2 M_{id} = \sum_i (X_i - WH) M_i (X_i - WH)^T = ||X - WH||_M^2$$

n'est pas équivalent à

$$\min J_2 = \sum_i (X_i - XWH) M_i (X_i - XWH)^T = ||X - XVH||_M^2$$

où $||Y||_M^2 = \sum_{i,j} Y_{ij}^2 M_{ij}$

L'interprétation courante de la transformation («factors and factor loadings») implique que l'on cherchera à utiliser le premier critère plutôt que le second. La dénomination PCA déformé est donc en ce sens érronée, puisque l'on ne cherche plus une projection orthogonale P approximant au mieux X par XPP^T mais plutôt à factoriser la matrice X, équivalent donc à NMF sans la contrainte de non négativité.

L'algorithme pour trouver W et H est très simple :

1. donner des valeurs aléatoires à H
2. minimiser par rapport à W connaissant H

$$W_{i.} \leftarrow X_{i.} M_i H (H M_i H)^{-1} \quad \text{avec} \quad M_i = \text{diag}(M_{i.})$$

3. minimiser par rapport à H connaissant W

$$H_{.d} \leftarrow (W^T M_d W)^{-1} M_d W^T X_{.d} \quad \text{avec} \quad M_d = \text{diag}(M_{.d})$$

4. retourner à l'étape 2 jusqu'à convergence

en effet, en annulant les dérivées partiels du critère par rapport à W_{ik} on trouve :

$$\begin{aligned} \frac{\partial J}{\partial W_{ik}} &= -2 \sum_d H_{kd} M_{id} (X_{id} - \sum_{k'} W_{ik'} H_{k'd}) = 0 \\ \nabla J(W_{i.}) &= -2(X_{i.} - W_{i.} H) M_i H^T = 0 \\ W_{i.} H M_i H^T &= X_{i.} M_i H^T \end{aligned}$$

L'algorithme converge dans le cas général vers un minimum local du critère. Pour vérifier l'existence de minimum locaux nous avons simplement relancé plusieurs fois l'algorithme avec les mêmes paramètres mais avec des initialisations différentes et observé des valeurs de J différentes.

26 NMF déformé

La formulation du problème NMF est assez très simple :

$$\min J = ||X - WH||^2 = \sum_i \sum_d (x_{id} - \sum_k w_{ik} h_{kd})^2$$
$$s.c \quad h_{kd} \geq 0 \; w_{ik} \geq 0$$

En introduisant des poids $M_i d$ on obtient :

$$\min J = \sum_i \sum_d (x_{id} - \sum_m w_{im} h_{md})^2 M_{id}$$
$$s.c \quad h_{md} \geq 0, \; w_{im} \geq 0$$

On ne donne pas une démonstration détaillée de la convergence de l'algorithme, en effet on reprend la démarche décrite dans « Algorithms for non-negative matrix factorization [8, Lee2001] », si dans les équations 10 à 25 de l'article on remplace W par $M_d^{1/2} W$ et V par $M_d^{1/2} X$, et par symétrie du problème sur W et H on trouve comme « update rules» :

$$\begin{aligned} H_{md} &\leftarrow H_{md} \frac{(W^T M_d X)_{md}}{(W^T M_d W H)_{md}} \\ W_{im} &\leftarrow W_{im} \frac{(X M_i H^T)_{im}}{(W H M_i H^T)_{im}} \end{aligned}$$

L'algorithme consiste simplement à initialiser W H avec des valeurs positives, et à appliquer itérativement et en séquence les « update rules » pour W et H.

27 Semi-NMF déformé

dans Convex and « Semi-Nonnegative Matrix Factorizations [19, Ding2006] », est introduit le problème de factorisation Semi-NMF.

$$\min J = ||X - WH||^2 = \sum_i \sum_d (x_{id} - \sum_k w_{ik} h_{kd})^2$$
$$s.c \quad h_{kd} \geq 0$$

En introduisant des poids $M_i d$ on obtient :

$$\min J = \sum_i \sum_d (x_{id} - \sum_k w_{ik} h_{kd})^2 M_{id}$$
$$s.c \quad h_{kd} \geq 0$$

Seule H est contrainte à être positive, W peut prendre des valeurs négatives.

On ne donne pas non plus une démonstration détaillée de la convergence de l'algorithme, en effet on reprend la démarche décrite dans l'article de Chris Ding, si dans les équations 8 à 17 on remplace W par $M_d^{1/2}W$ on trouve comme « update rules » :

$$H_{kd} \leftarrow H_{kd}\sqrt{\frac{(W^T M_d X)^+_{kd} + [(W^T M_d W)^- H]_{kd}}{(W^T M_d X)^-_{kd} + [(W^T M_d W)^+ H]_{kd}}}$$
$$W_{i.} \leftarrow X_{i.} M_i H^T (H M_i H^T)^{-1}$$

avec $A^+ = \max(A, 0)$, $A^- = -\min(A, 0)$, $A = A^+ - A^-$.

On peut également inverser le rôle de W et H, le problème étant essentiellement symétrique.

Comme pour l'algorithme NMF donné plus haut, l'algorithme consiste simplement à initialiser W H avec des valeurs non nulles respectant les contraintes, et à appliquer itérativement les « update rules » pour W et H.

28 Fischer / Linear Discriminant Analysis

Il s'agit d'une technique projection ou de réduction de dimensionnalité supervisée.

soit $N = \sum_c N_c$ points répartis en C classes, chacune de centroïde $\mu_c = \frac{1}{N_c}\sum_{i\in c} x_i$ et de matrice d'autocovariance intra-classe Σ_c. On considère également $\mu = \frac{1}{N}\sum$ le centroïde total.

soit $\Sigma_W = \frac{1}{C}\Sigma_c$ la moyenne des matrices d'autocovariance intra-classe

et $\Sigma_B = \frac{1}{C}\sum_c (\mu_c - \mu)(\mu_c - \mu)^T$ la matrice de variance inter classe.

le « Fisher criterion » que l'on cherche à maximiser est :

$$J = \frac{tr(\Sigma_B)}{tr(\Sigma_W)}$$

en projetant orthogonalement les données sur un sous-espace de dimension K défini par une matrice v on trouve

$$J(v) = \frac{tr(v^T \Sigma_B v)}{tr(v^T \Sigma_W v)}$$

Le sous-espace de dimension K qui maximise $J(v)$ est alors défini par la matrice $v = [v_1 \cdots v_K]$ dont les colonnes sont les K vecteurs propres correspondant aux K plus grandes valeurs propres de $\Sigma_W^{-1}\Sigma_B$.

Généralement on choisit K comme étant égal à $C - 1$, et les vecteurs propres correspondant aux seules $C - 1$ valeurs propres non nulles de $\Sigma_W^{-1}\Sigma_B$.

Projeter les données dans ce sous-espace peut faire perdre de l'information qui aurait pu être fortement discriminante après une transformation non linéaire, puisque ce sont les K descripteurs mélanges linéaires les plus linéairement discriminant qui sont gardés.

On préfère dans certain cas maximiser une variante du « Fisher criterion » :

$$J = \frac{tr(\Sigma_B)}{tr(\Sigma_W) + tr(\Sigma_B)} = \frac{tr(\Sigma_B)}{tr(\Sigma)}$$

Dans ce cas le sous-espace où l'on projetera les données est défini par les K premier vecteurs propres de $\Sigma^{-1}\Sigma_B$.

29 Principal Component Analysis

29.1 suppositions probabilistes

soit x un vecteur aléatoire réel centré de dimension D dont les composantes sont corrélées et R_{xx} sa matrice d'autocovariance. Celle-ci est définie positive et diagonalisable dans une base réelle orthonormée avec des valeurs propres positives.

on écrit que $R_{xx} = P\Lambda P^T$ avec les valeurs propres $\lambda_1 \cdots \lambda_D$ rangées par ordre décroissant

soit $X : N \times D$ une matrice d'observations, μ_X la moyenne empirique des colonnes de X et $Y = (X - \mu_X)P$ et $\widehat{R_{YY}} = \frac{1}{N-1}Y^T Y$ on a :

on rappelle que $\widehat{R_{XX}} = \frac{1}{N-1}(X - \mu_X)^T(X - \mu_X)$ est un estimateur non biaisé de la matrice d'autocovariance de x.

$$\mathbb{E}(\widehat{R_{YY}}) = \mathbb{E}(\frac{1}{N-1}Y^TY) = \mathbb{E}(\frac{1}{N-1}P^T(X-\mu_X)^T(X-\mu_X)P) = P^T\mathbb{E}(\widehat{R_{XX}})P = P^TR_{XX}P = \Lambda$$

on a donc bon espoir qu'après rotation de l'espace par P les composantes des observations transformées soient décorrélées et de variance maximale dans les premiers coefficients

29.2 démonstration empirique de l'optimalité de la transformation PCA

soit $X : N \times D$ une matrice d'observations, une dimension $K < D$, et on impose que X soit de rang supérieur ou égal à K.

On cherche une matrice $Q : D \times K$, une matrice $R : K \times D$ et un vecteur $\mu : 1 \times D$ qui minimise :

$$J = \sum_{n=1}^{N} ||x_n - (x_n - \mu)QR - \mu||^2$$

en calculant le gradient de J par rapport à μ on trouve :

$$\nabla\mu = \sum_{n=1}^{N}(QR - I)(x_n - (x_n - \mu)QR - \mu)$$

en posant $\nabla\mu = 0$ on trouve :

$$N(QR - I)\mu = (QR - I)\sum_{n=1}^{N} x_n$$

si bien que l'on peut choisir :

$$\mu = \frac{1}{N}\sum_{n=1}^{N} x_n$$

on pose donc

$$\bar{x}_n = x_n - \mu$$

$$J = \sum_{n=1}^{N} ||\bar{x}_n - \bar{x}_nQR||^2$$

Soit E le sous espace engendré par QR, QR doit être une projection orthogonale sur ce sous-espace, et celui-ci doit être de dimension maximale K. (une justification possible de ce fait est donnée dans le paragraphe suivant 29.2.1).

Par définition d'une projection orthogonale, on peut écrire qu'il existe $QR = PP^T$, et $P^TP = I_K$, et par la suite :

$$\min J = \min ||\bar{X} - \bar{X}PP^T||^2 \Leftrightarrow \min tr(\bar{X}PP^TPP^T\bar{X}^T) - 2tr(\bar{X}PP^T\bar{X}^T) \Leftrightarrow \max tr(\bar{X}P^TP\bar{X}^T)$$

en utilisant le fait que $tr(AB) = tr(BA)$, et en posant $\Sigma = \bar{X}\bar{X}^T = V\Lambda V^T$ avec les valeurs propres λ_n rangées par ordre décroissant, on trouve :

$$tr(\bar{X}P^TP\bar{X}^T) = tr(P\bar{X}^T\bar{X}P^T) = tr(PV\Lambda V^TP^T)$$

ce qui donne la solution :

$$P = [V_1, \cdots, V_K]$$

soit les K propres vecteurs associés aux K plus grandes valeurs propres de $\bar{X}\bar{X}^T$

29.2.1 autre formulation du problème PCA

soit $X : N \times D$ une matrice d'observations centrées, une dimension $K < D$, et on impose que X soit de rang supérieur ou égal à K.

$W : N \times K$ et $H : K \times D$

$$J_2 = ||X - WH||^2$$

est minimisé lorsque WH est de rang K et $W = XH^T(HH^T)^{-1}$. minimiser J_2 est donc équivalent à minimiser J (avec $W = XQ$ et $H = R$) et donc à résoudre PCA.

Cela suggère un algorithme de type « alternated least square » qui nous sera utile pour proposer des généralisations de PCA :

1. initialiser les matrices W et H
2. optimiser W connaissant H

$$W \leftarrow XH^T(HH^T)^{-1}$$

3. optimiser H connaissant W

$$H \leftarrow (W^TW)^{-1}W^TX$$

4. retourner à l'étape 2 jusqu'à convergence

29.3 Les centroïdes obtenus par clustering K-means sont dans l'espace PCA

Dans « K-means Clustering via Principal Component Analysis » [20, Ding2004] une démonstration de ce théorème est donnée, nous n'avons malheureusement pas réussi à la résumer et à la rédiger ici, mais il nous parait intéressant de vérifier si ce théorème reste vrai dans le cas des K-means déformés, après une factorisation telle que celle donnée au paragraphe 25. Le théorème y est énnoncé de la manière suivante :

Les centroïdes optimaux pour K-means appartiennent au sous-espace défini par les $K-1$ premiers vecteurs propres de la matrice d'autocorrélation, soit les $K-1$ premières composantes principales.

30 Non-negative Matrix Factorization

voir NMF déformé 26.

31 Kernel Principal Component Analysis

rappelons que PCA est équivalent à trouver une base orthonormée dans laquelle la matrice de covariance $\widehat{R_{XX}} = \frac{1}{M-1}\sum_{n=1}^{N} x_n^T x_n$ est diagonale (en supposant les observations centrées empiriquement)
c'est à dire résoudre :

$$\lambda v^T = \frac{1}{M-1}\sum_{n=1}^{N} x_n^T x_n v^T \quad (1)$$

en posant $\alpha_i = \frac{x_i v^T}{\lambda(M-1)}$ on voit que

$$v^T = \sum_{i=1}^{N} \alpha_i x_i^T$$

donc que les vecteurs propres peuvent s'écrire comme combinaisons linéaires des observations
ce qui permet de réécrire (1) :

$$\forall k \quad \lambda x_k v^T = \frac{1}{M-1}\sum_{n=1}^{N} x_k x_n^T x_n v^T \quad (2)$$

ou encore en remplaçant v^T par son expression :

$$\forall k \quad \lambda x_k \sum_{i=1}^{N} \alpha_i x_i^T = \frac{1}{M-1}\sum_{n=1}^{N} x_k x_n^T x_n \sum_{i=1}^{N} \alpha_i x_i^T \quad (3)$$

enfin en écrivant $K(i,j) = x_i x_j^T$ on obtient :

$$\forall k \quad \lambda \sum_{i=1}^{N} \alpha_i K(k,i) = \frac{1}{M-1}\sum_{n=1}^{N} K(k,n) \sum_{i=1}^{N} \alpha_i K(n,i) \quad (4)$$

ce qui donne en écriture matricielle :

$$(M-1)\lambda K\alpha = KK\alpha \quad (5)$$

$$\lambda\alpha = \frac{1}{M-1}K\alpha \quad (6)$$

les solutions de l'équation (6) donnant immédiatement les solutions de (5)

il suffit donc de diagonaliser $\frac{1}{M-1}K$ (qui est diagonalisable dès que l'on suppose qu'elle est une matrice de Gram) et de normaliser les vecteurs propres α obtenus pour que v soit de norme 1

$$v^T v = \sum_{i=1}^{N} \alpha_i x_i^T \sum_{i=1}^{N} \alpha_i x_i = \alpha K \alpha^T = 1$$

toutes ces équations restent correctes en appliquant une transformation non linéaire ϕ aux observations : $K(i,j) = \phi(x_i)\phi(x_j)^T$ mais la supposition que les points sont centrés n'est plus forcément respectée.

si $\tilde{K}$ est la matrice de Gram des points centrés, on peut écrire que

$$\tilde{K}(i,j) = (\phi(x_i) - \frac{1}{N}\sum_{n=1}^{N}\phi(x_n))(\phi(x_j) - \frac{1}{N}\sum_{n=1}^{N}\phi(x_n))^T = K(i,j) - \frac{1}{N}\sum_{n=1}^{N}K(i,n) - \frac{1}{N}\sum_{n=1}^{N}K(n,j) + \frac{1}{N^2}\sum_{n=1}^{N}\sum_{m=1}^{N}K(n,m)$$

soit en écriture matricielle :

$$\tilde{K} = K - K1_N^T - 1_N K + 1_N K 1_N^T$$

31.1 Formulation explicite des points dans l'espace de redescription (Empirical Feature Space)

si Kernel PCA permet de se faire une idée de la position de nos points dans l'espace de resdescription, une formulation explicite est préférable.

soit $K(x,y)$ un noyau défini positif. La matrice de Gram M formée à partir de K et des observations x_i est définie positive. Ainsi elle peut être diagonalisée dans une base orthonormée avec des valeurs propres positives.

$$M = P\Lambda P^T$$

on suppose qu'il existe Y où chaque ligne Y_i contient l'image dans l'espace de resdescription d'une observation x_i, on a donc

$$M = YY^T$$

et une solution possible est

$$Y = P\Lambda^{1/2}P^T$$

ainsi l'espace de redescription est de dimension N le nombre d'observation.

grâce à cette formulation explicite on peut transformer une matrice d'observation à l'aide d'une fonction noyau (définie positive). Dans le cas du noyau gaussien cela consiste à projeter les points sur une hypersphère de dimension N ce qui permet ensuite d'appliquer tous les algorithmes habituels linéaires.

cette transformation montre de manière intuitive que la mesure de similarité réalisée dans l'espace originel devient une mesure de corrélation dans l'espace transformé

$$e^{-||x_i - x_j||^2} = \frac{y_i . y_j}{||y_i||\ ||y_j||}$$

ce qui est intéressant c'est que l'on peut projeter de nouveaux points $x, x', \cdots$ dans cet espace, en calculant $K(x, x_i)$ pour chaque x_i utilisé pour construire l'espace, et en projetant de la manière suivante :

$$\phi(x) = [K(x,x_1), \cdots K(x,x_N)]K^{-1/2}$$

$$K^{-1/2} = P\Lambda^{-1/2}P^T$$

Si une valeur propre est nulle le système est sous-déterminé et il est donc préférable de s'assurer que les points x_i forment bien une matrice de similarité de rang plein.

On aura donc $\phi(x).y_i = K(x,x_i)$ mais pour deux nouveaux points x, x' projetés on n'aura pas $\phi(x).\phi(x') = K(x,x')$

référence : [24, Scholkopf1999]

32 MFCC et ondelettes

Les MFCC et les ondelettes de Haar ont entre autre été utilisés pour des essais d'optimisation de pondération de feature, mais peu de résultat très concluant en sont ressortis pour le moment.

33 Generalized EM pour réaliser un clustering en classes d'enveloppes avec modèle de source multiplicatif

Generalized EM désigne une version modifée d'EM dans le cas où on ne sait pas minimiser $Q(\Theta, \Theta^n)$ à chaque itération. Dans ce cas, on peut à chaque itération faire converger $Q(\Theta, \Theta^n)$ vers un maximum local (en partant de $\Theta = \Theta^n$), ou simplement trouver un Θ tel que $Q(\Theta, \Theta^n) > Q(\Theta^n, \Theta^n)$. Dans ce cas le (log-)likelihood convergera comme avec EM vers un maximum local (suivant l'initialisation et le type d problème, celui-ci pourra être plus ou moins éloigné du maximum global).

A_i est un vecteur contenant les amplitudes de partiels de la note i, M_i est la matrice indicatrice des fréquences des partiels. $\Theta = \{\{\pi_1, e_1, D_1\}, \cdots \{\pi_C, e_C, D_C\}\}$ représente les paramètres à estimer, z_{ic} les « missing values » et x_i représente les données connues.

$$p(x_i|z_{ic} = 1, \Theta) = p(A_i, M_i|e_c, D_c, \sigma_c) = \frac{1}{\sqrt{2\pi}\sigma_c} e^{-\frac{||A_i - e_c H M_i D_c||^2}{2\sigma_c^2}}$$

$$p(z_{ic} = 1|\Theta) = \pi_c$$

$$p(X|\Theta) = \prod_i p(x_i|\Theta) = \prod_i \sum_c \pi_c p(A_i, M_i|e_c, D_c, \sigma_c)$$

On reprend les démarches décrites dans [22, EM] et [15, gmmEM], on trouve alors :

- E-step

$$\bar{z_{ic}} = p(z_{ic} = 1|x_i, \Theta_n) = \frac{p(x_i|z_{ic} = 1, \Theta_n)\pi_c}{p(x_i|\Theta_n)}$$

- M-step
  choisir Θ^{n+1} tel que

$$Q(\Theta^{n+1}, \Theta^n) > Q(\Theta^n, \Theta^n)$$

$$\begin{aligned}
Q(\Theta, \Theta^n) &= \sum_i \mathbb{E}_{z_i|x_i,\Theta^n} \log p(x_i, z_i|\Theta) \\
&= \sum_i \sum_c \bar{z_{ic}} \log p(x_i, |z_{ic} = 1\Theta)\pi_c \\
&= \sum_i \sum_c \bar{z_{ic}} \log p(x_i, |e_c, D_c, \sigma_c)\pi_c \\
&= \sum_i \sum_c \bar{z_{ic}} \left(\log(\pi_c) - \log(\sqrt{2\pi}\sigma_c) - \frac{||A_i - e_c H M^i D_c||^2}{2\sigma_c^2} \right)
\end{aligned}$$

on peut choisir :

$$D_{cp}^{n+1} = \sum_i \bar{z_{ic}} \frac{A_{ip}}{e_c H M_{p.}^i}$$

$$R_c = H \sum_i \bar{z_{ic}} M^i D_c (M^i)^T H^T$$

$$B_c = \sum_i \bar{z_{ic}} A_i D_c (M^i)^T H^T$$

$$e_c^{n+1} = e_c^n (B_c R_c^T)./(e_c^n R_c R_c^T)$$

$$\sigma_{n+1}^2 = \frac{\sum \bar{z_{ic}} ||A_i - e_c H M^i D_c||^2}{\sum \bar{z_{ic}}}$$

34 Generalized EM pour le modèle de source multiplicative

Generalized EM désigne une version modifée d'EM si on ne sait pas minimiser $Q(\Theta, \Theta^n)$ à chaque itération. Dans ce cas, on peut à chaque itération faire converger $Q(\Theta, \Theta^n)$ vers un maximum local (en partant de $\Theta = \Theta^n$), ou simplement trouver un Θ tel que $Q(\Theta, \Theta^n) > Q(\Theta^n, \Theta^n)$. Le (log-)likelihood convergera comme avec EM vers un maximum local (suivant l'initialisation et le type de problème, celui-ci pourra être plus ou moins éloigné du maximum global).

x_i est un vecteur contenant les amplitudes de partiels de la note i, Π_i est la matrice indicatrice des fréquences des partiels. $\Theta = \{\{\pi_1, e_1, D_1, \sigma_1\}, \cdots \{\pi_C, e_C, D_C, \sigma_C\}\}$ représente les paramètres à estimer, z_{ic} les « missing values » et $y_i = \{x_i, \Pi_i\}$ représente les données connues.

$$p(y_i|z_{ic} = 1, \Theta) = p(x_i, \Pi_i|e_c, D_c, \sigma_c) = \frac{1}{\sqrt{2\pi}\sigma_c} e^{-\frac{||x_i - e_c H \Pi_i D_c||^2}{2\sigma_c^2}}$$

$$p(z_{ic} = 1|\Theta) = \pi_c$$

il serait naturel de considérer :

$$\mathcal{L}(\Theta; Y) = p(Y|\Theta) = \prod_i p(y_i|\Theta) = \prod_i \sum_c \pi_c p(x_i, \Pi_i|e_c, D_c, \sigma_c)$$

mais il est plus commode de remplacer $\mathcal{L}(\Theta, X)$ par une version régularisée, par exemple en ajoutant un a priori sur les paramètres du modèle :

$$\mathcal{L}_R(\Theta, X) = \mathcal{L}(\Theta, X)\mathcal{P}(\Theta)$$

on choisit naturellement :

$$\mathcal{P}(\Theta) = \exp(-\gamma||e||^2 - \delta||D||^2)$$

Avec $\gamma, \delta > 0$. Par ailleurs on impose la contrainte que $D \geq a$ pour un certain $a > 0$.

On reprend les démarches décrites dans [22, EM] et [15, gmmEM], on trouve alors :

- E-step
  les termes de régularisation ne rentrent pas en jeu :

$$\bar{z_{ic}} = p(z_{ic} = 1|y_i, \Theta_n) = \frac{p(y_i|z_{ic} = 1, \Theta_n)\pi_c}{p(y_i|\Theta_n)}$$

- M-step
  choisir Θ^{n+1} tel que :

$$Q(\Theta^{n+1}, \Theta^n) > Q(\Theta^n, \Theta^n)$$

$$\begin{aligned}
Q(\Theta, \Theta^n) &= -\gamma||e||^2 - \delta||D||^2 + \sum_i \mathbb{E}_{z_i|y_i,\Theta^n} \log p(x_i, z_i|\Theta) \\
&= -\gamma||e||^2 - \delta||D||^2 + \sum_i \sum_c \bar{z_{ic}} \log p(y_i, |z_{ic} = 1\Theta)\pi_c \\
&= -\gamma||e||^2 - \delta||D||^2 + \sum_i \sum_c \bar{z_{ic}} \log p(x_i, \Pi_i|e_c, D_c, \sigma_c)\pi_c \\
&= -\gamma||e||^2 - \delta||D||^2 + \sum_i \sum_c \bar{z_{ic}} \left(\log(\pi_c) - \log(\sqrt{2\pi}\sigma_c) - \frac{||x_i - e_c H \Pi_i D_c||^2}{2\sigma_c^2}\right)
\end{aligned}$$

on peut choisir (avec $a > 0$) :

$$e_c^{n+1} = \sum_i z_{ic} x_i D_c \Pi_i^T (H(\sum_i z_{ic} \Pi_i D_c^2 \Pi_i^T) H + \gamma I)^{-1}$$

$$d_c(p)^{n+1} = \max(a, (\sum_{i \in c} z_{ic} e_c H \Pi_i(., p) x_i(p))/(\sum_{i \in c} z_{ic} (e_c H \Pi_i(., p))^2 + \delta))$$

$$\sigma_c^{n+1} = \sqrt{\frac{\sum \bar{z_{ic}} ||x_i - e_c H \Pi_i D_c||^2}{\sum \bar{z_{ic}}}}$$

35 EM pour le modèle de source additive

On reprend les même démarches que dans le paragraphe précédant, en s'appuyant cette fois sur le modèle avec source additive, et l'algorithme K-means utilisant ce modèle (voir paragraphe 19.5). On sait minimiser la fonction auxiliaire Q à chaque itération donc il s'agit bien un algorithme EM et non plus GEM. Comme l'algorithme K-means, il converge vers un maximum local du likelihood, selon l'initialisation utilisée.